DTiF
Digital Technologies in focus
Initiative of and funded by the Australian Government Department of Education and Training

acara AUSTRALIAN CURRICULUM, ASSESSMENT AND REPORTING AUTHORITY

CLASSROOM IDEAS: YEARS 6–8

MICRO:BIT MISSIONS: TAKE A CHANCE ON ME (INTEGRATING MATHEMATICS)

The micro:bit is a small, programmable microcontroller that can be used to teach Digital Technologies knowledge and understanding and skills. The micro:bit can be coded for different purposes. Data can be inputted and outputted in various ways including through the user interface (Figures 1 and 2) LED display and buttons.

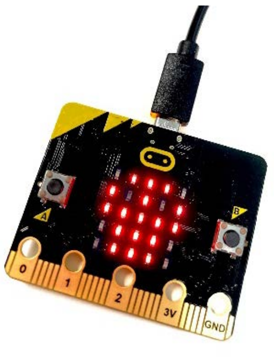


Figure 1: A micro:bit showing a representation of a skull with LED lights

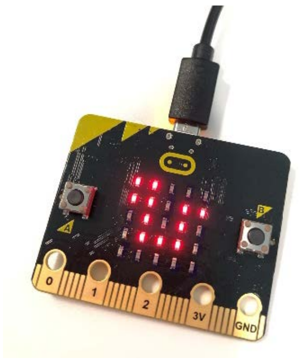


Figure 2: A micro:bit showing a representation of a snake with LED lights

This resource comprises two activities that can be used together or if you prefer as individual activities to explore the concept of chance in Mathematics. Students will use computational thinking and learn about Digital Technologies through exploring and using the micro:bit as a digital system to generate and collect data and implement and/or modify a program (algorithm) involving branching, iteration and functions in visual and/or general-purpose programming language.

Real-world application: Using the concept of randomisation to generate data or design a digital device has application in a range of industries and the creative arts. For example, in a music playlist, a random number generator can be used to shuffle songs. Scientists use randomised controlled trials to test hypotheses. Randomness is also used in gaming and in lotteries and raffles. Read more in Leong, T, Howard, S & Vetere, F 2008, 'Take a chance on me: using randomness for the design of digital devices', *Interactions – Optimistic futurism,* vol. 15, issue 3, May + June, pp. 16–19. DOI 10.1145/1353782.135378. Access at https://people.eng.unimelb.edu.au/showard/papers/Int08.pdf

Resources required

- micro:bits and cables (one per student, group or station)
- www.makecode.org website or Python coding editor, for example Mu https://codewith.mu/

Mission: To demonstrate the ability to generate and interpret random data

These activities give students an opportunity to explore probability or the likelihood of something happening. To do this, students need to collect and interpret data. The micro:bit will be programmed to mimic two chance situations:

- flipping a coin (heads or tails)
- rolling a dice. Note: two versions are included: digital dice roll (version 1) and quick digital dice roll (version 2).

Each situation will generate data that can be collated, providing opportunities for visual presentation of the data (Years 5 and 6) and data representation (Years 7 and 8). For the quick dice roll, the micro:bit will be programmed to very quickly roll a dice 100 times and report back how many times each number 1–6 is rolled. This is simply to demonstrate how quickly the micro:bit can achieve a task and store the results in an array.

For more information about how arrays store data in digital systems see: https://makecode.microbit.org/courses/csintro/arrays/overview

Note: Mathematically, an array represents a multiplicative relationship; therefore, it is better to use a table of data and discuss the reason for this if the activity has a mathematical focus.

This activity provides an ideal opportunity for students to:

- explore the micro:bit as a digital system including hardware and software components. For example draw an annotated diagram of the micro:bit as a digital system using labels of inputs and outputs together with the software and hardware involved
- explicitly discuss the computational thinking required in the activity
- plan the method they will use to collect and visualise data from the activity.

Activity instructions

1. Set up each activity station using the equipment required.
2. Choose the most appropriate coding algorithm for your context and student ability. The algorithms are provided in:
   A. pseudocode (simple terms or plain English) – page 4
   B. block code (visual programming language) suited to Years 5–6 – page 5
   C. Python (general-purpose programming language) suited to Years 7–8 – page 7.
3. Once the micro:bit has been coded, follow the user algorithm on the next page.
4. To model the functions of a digital system, choose a method to collect and interpret the data.
   Optional: You could design an algorithm to have the micro:bit collect the data for you. An example of this is shown in Figure 3.

User algorithm

Coin toss

1. Take your micro:bit – make sure it has a battery attached.
2. Press button A.
3. Record what comes up (heads or tails).
4. Repeat 20 times.
5. Graph your results (an example is shown in Figure 4).

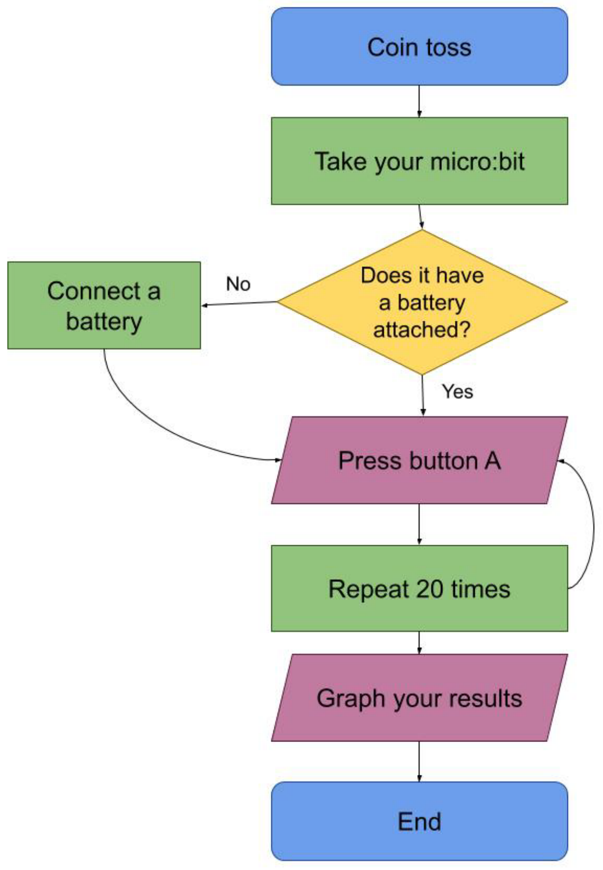
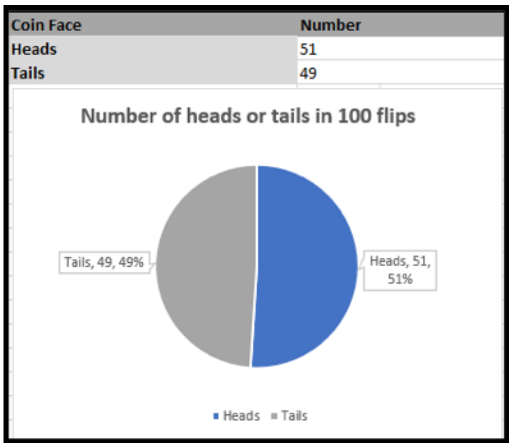


Figure 3: A visual algorithm of the coin toss



Coin Face	Number
Heads	51
Tails	49

Figure 4: One-way students may graph the results of a coin toss. Students could compare results from 20, 100 and a greater number

Digital dice (for either version 1 or 2)

1. Press button B.
2. Record the number that comes up.
3. Repeat 20 times.
4. Graph your results.

Coding algorithms

A. Pseudocode (simple terms or plain English)

Coin toss

```
START when button A is pressed
   Choose a random state (True or False)
   IF True
         Then display heads (skull)
   ELSE
         Display tails (snake)
END
```

Optional additional coin toss pseudocode

Consider adding code like this if you would like to have the micro:bit flip the coin and count the number of heads and tails for you. Students could record the number of heads and tails for a small number of flips and then progressively larger numbers of flips. They should notice the ratio converge to 1.

```
FOREVER
   WHILE Button A is not pressed
      Pick a random number between 0 and 1
      IF it's 0 SET Heads = Heads+1
      ELSE SET Tails = Tails +1

On Button A pressed
   Show number Heads
   Pause 1000
   Show number Tails
```

Digital dice roll (version 1)

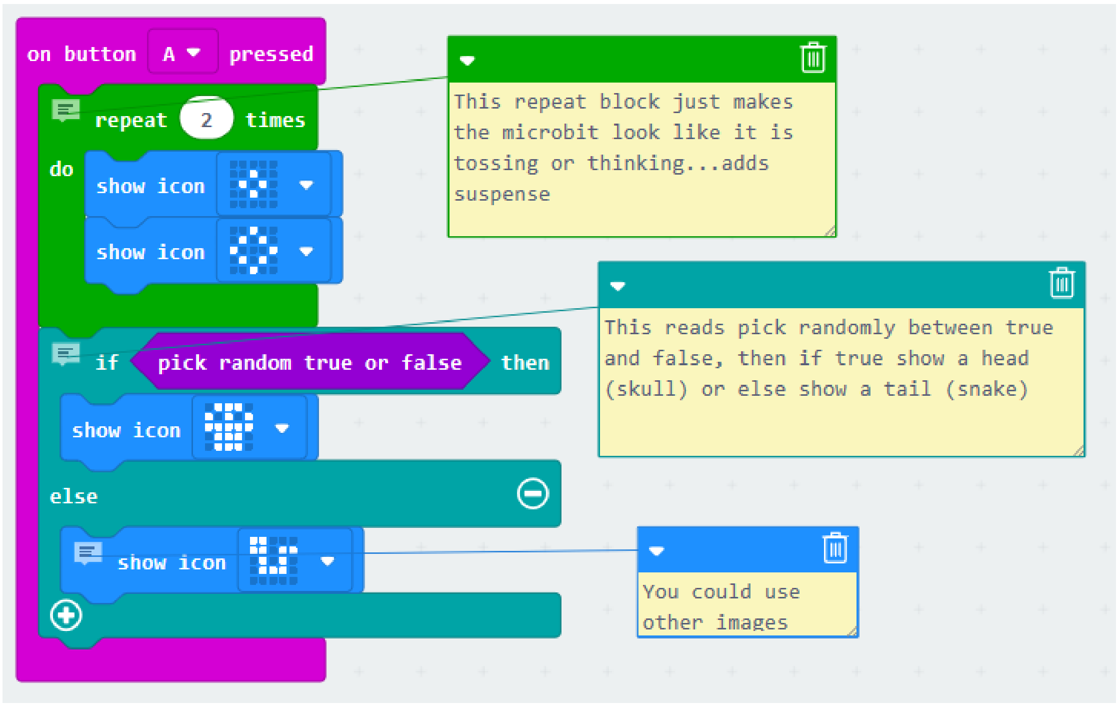
```
Begin when button B is pressed
          Choose a random number between 1 and 6 and store it in a variable (called random)
          Display random number
End
```

Quick digital dice roll (version 2)

```
Begin when buttons A + B are pressed together
Create a variable called counter and set its value to 1
Create a variable for each of the numbers 1–6 (one, two, three, four, five, six) and set them all to zero
Repeat
Display small square (to show it is working) … If you want it to work fast don't display either square
Choose a random number between 1 and 6
    If random number is 1
        Add 1 to one
    Else if random number is 2
        Add 1 to two
    Else if random number is 3
        Add 1 to three
    …
    Else if random number is 6
        Add 1 to six
    Add 1 to counter
    Display large square (to show it is working)
Until counter = 100
Display one, two, three…six (One – 12, Two – 8, Three – 15…Six – 11)
```

B. Block code (visual programming language)

Coin toss



```
on button A pressed
    repeat 2 times
    do
        show icon
        show icon
    if pick random true or false then
        show icon
    else
        show icon
```

This repeat block just makes the microbit look like it is tossing or thinking...adds suspense

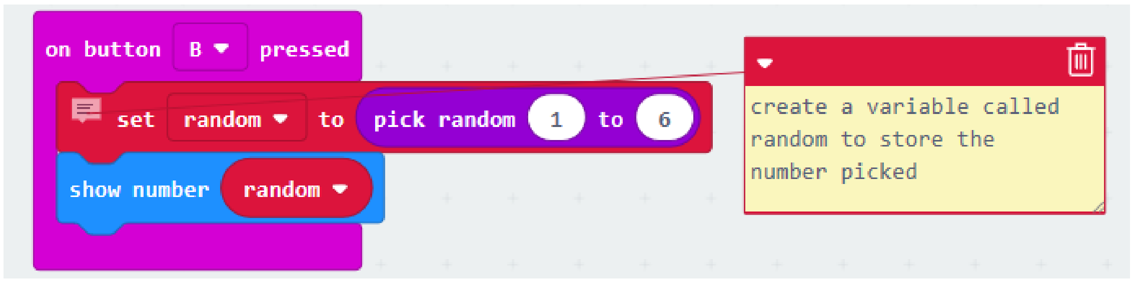
This reads pick randomly between true and false, then if true show a head (skull) or else show a tail (snake)

You could use other images

Figure 5

Figures 5 and 6 show comments on yellow note cards to the right of the code blocks. These can be used by a teacher to describe what is happening in the code itself to scaffold learning or to provide feedback. They can also be used by a student to describe how they have designed and implemented code.

Digital dice roll (version 1)



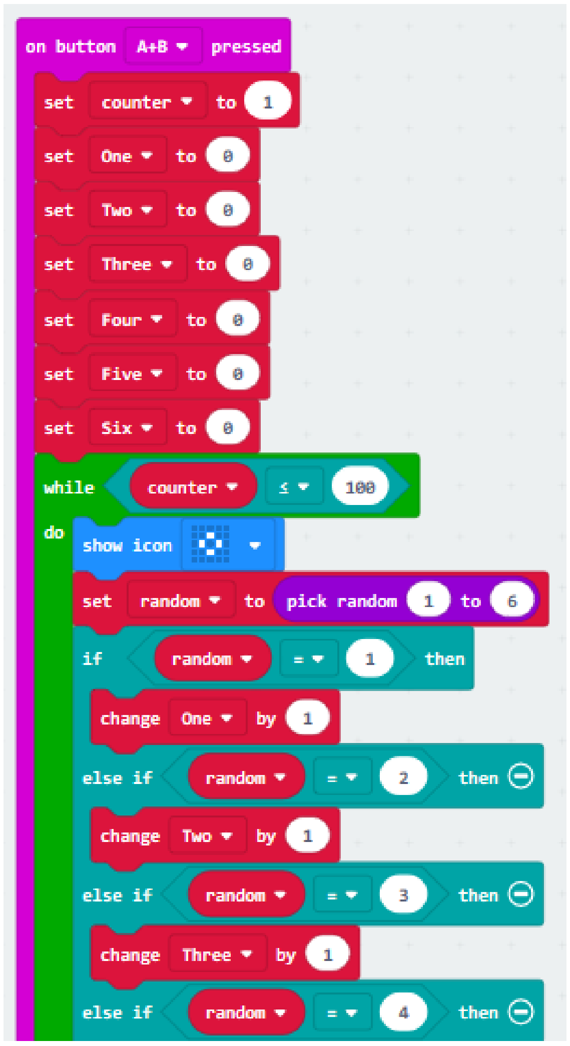
```
on button B pressed
    set random to pick random 1 to 6
    show number random
```

```
create a variable called
random to store the
number picked
```

Figure 6

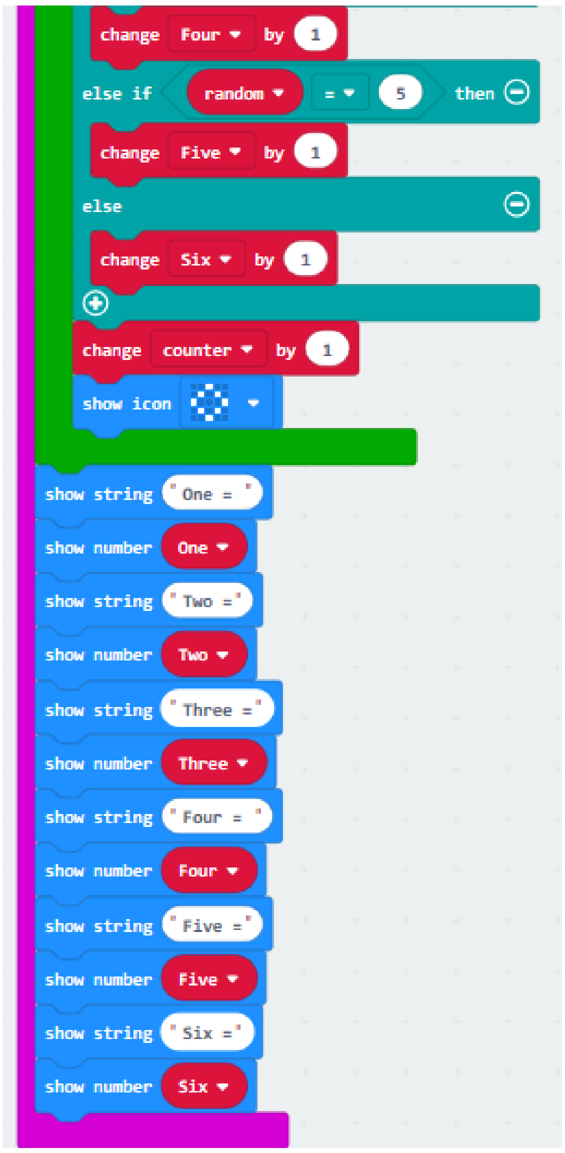
Quick digital dice roll (version 2)

The quick digital dice roll version 2 is demonstrated in both Figures 7a and 7b.



```
on button A+B pressed
    set counter to 1
    set One to 0
    set Two to 0
    set Three to 0
    set Four to 0
    set Five to 0
    set Six to 0
    while counter ≤ 100
    do
        show icon
        set random to pick random 1 to 6
        if random = 1 then
            change One by 1
        else if random = 2 then
            change Two by 1
        else if random = 3 then
            change Three by 1
        else if random = 4 then
```

Figure 7a



```
            change Four by 1
        else if random = 5 then
            change Five by 1
        else
            change Six by 1
        change counter by 1
        show icon
    show string "One ="
    show number One
    show string "Two ="
    show number Two
    show string "Three ="
    show number Three
    show string "Four ="
    show number Four
    show string "Five ="
    show number Five
    show string "Six ="
    show number Six
```

Figure 7b

C. Python code (general-purpose programming language)

The comments indicated in green and preceded by a hashtag (#) are not part of the code and are there only to further explain the code in more detail.

Coin toss

```
from microbit import *                      # get the standard microbit library
import random                               # get lib that generates random numbers

while True:                                 # do this forever
    if button_a.was_pressed():
        toss = random.randrange(1, 3)       # chooses a random number between 1 & 2
        if toss == 1:                       # note the double == to mean equals
            display.show(Image.SKULL)       # skull indicates heads
        else:
            display.show(Image.SNAKE)       # snake indicates tails
        sleep(1000)
    display.clear()
```

Digital dice roll (version 1)

```
from microbit import *                    # get the standard micro:bit library
import random                             # get lib that generates random numbers

while True:                               # do this forever
    if button_b.was_pressed():
        dice = random.randrange(1, 7)     # chooses a random number between 1 & 6
        display.show(dice)                # show result of the random selection
        sleep(1000)
```

Quick digital dice roll (version 2)

```
from microbit import *                    # get the standard micro:bit library
import random                             # get lib that generates random numbers

# list for quick coin toss
mylist = []    # create an empty list
while True:
    if button_a.was_pressed():
        for i in range(20):   # repeat the below line 20 times
            mylist.append(random.randrange(1, 7))  # add rnd number (1-6) to list

        one = mylist.count(1)  # find how many 1s in the list
        two = mylist.count(2)   # find how many 2s in the list
        three = mylist.count(3)
        four = mylist.count(4)
        five = mylist.count(5)
        six = mylist.count(6)

        display.scroll(one)  # print the number of 1s
        display.scroll(two)  # print the number of 2s
        display.scroll(three)
        display.scroll(four)
        display.scroll(five)
        display.scroll(six)
```

Combine all chance programs into one

Below is code that combines the three chance programs into one program.

```
from microbit import *                      # get the standard micro:bit library
import random                               # get lib that generates random numbers


while True:   # do this forever
    if button_a.was_pressed():
        toss = random.randrange(1, 3)       # chooses a random number between 1 & 2
        if toss == 1:                       # note the double == to mean equals
            display.show(Image.SKULL)       # skull indicates heads
        else:
            display.show(Image.SNAKE)       # snake indicates tails
        sleep(1000)
    display.clear()

    if button_b.was_pressed():
        dice = random.randrange(1, 7)       # chooses a random number between 1 & 6
        display.show(dice)                  # show result of the random selection
        sleep(1000)

    if accelerometer.was_gesture('left'):  # move micro:bit left to activate
        mylist = []  # create a blank list
        for i in range(20):   # repeat the below line 20 times
            mylist.append(random.randrange(1, 7))  # add rnd number (1-6) to list

        one = mylist.count(1)  # find how many 1s in the list
        two = mylist.count(2)   # find how many 2s in the list
        three = mylist.count(3)
        four = mylist.count(4)
        five = mylist.count(5)
        six = mylist.count(6)

        display.scroll(one)  # print the number of 1s
        display.scroll(two)  # print the number of 2s
        display.scroll(three)
        display.scroll(four)
        display.scroll(five)
        display.scroll(six)
```

Links to the Australian Curriculum

Tables 1 and 2 give teachers an opportunity to see related aspects of the Australian Curriculum.

Table 1: Aspects of the Australian Curriculum: Digital Technologies 5–6 which may be addressed depending on the task.

Digital Technologies ***Achievement standard***	By the end of Year 6, students explain the fundamentals of digital system components (hardware, software and networks) and how digital systems are connected to form networks. They explain how digital systems use whole numbers as a basis for representing a variety of data types. Students define problems in terms of data and functional requirements and design solutions by developing algorithms to address the problems. They incorporate decision-making, repetition and user interface design into their designs and implement their digital solutions, including a visual program. They explain how information systems and their solutions meet needs and consider sustainability. Students manage the creation and communication of ideas and information in collaborative digital projects using validated data and agreed protocols.		
Strands	Digital Technologies knowledge and understanding • Digital systems Digital Technologies processes and production skills • Collecting, managing and analysing data • Creating digital solutions by: – Investigating and defining – Generating and designing – Producing and implementing		
Content descriptions	• Examine the main components of common digital systems and how they may connect together to form networks to transmit data (ACTDIK014)* • Acquire, store and validate different types of data, and use a range of software to interpret and visualise data to create information (ACTDIP016)* • Define problems in terms of data and functional requirements drawing on previously solved problems (ACTDIP017) • Design, modify and follow simple algorithms involving sequences of steps, branching, and iteration (repetition) (ACTDIP019) • Implement digital solutions as simple visual programs involving branching, iteration (repetition), and user input (ACTDIP020)		
Key concepts	• data collection* • data interpretation • specification • algorithms • implementation • digital systems	***Key ideas***	Thinking in Technologies • computational thinking
Cross-curriculum priorities		***General capabilities***	• Information and Communication Technology (ICT) Capability • Literacy • Numeracy

*aspects which may or may not be addressed depending on the lesson design.

Table 2: Aspects of the Australian Curriculum: Digital Technologies 7–8 which may be addressed depending on the task.

Digital Technologies ***Achievement standard***	By the end of Year 8, students distinguish between different types of networks and defined purposes. They explain how text, image and audio data can be represented, secured and presented in digital systems. Students plan and manage digital projects to create interactive information. They define and decompose problems in terms of functional requirements and constraints. Students design user experiences and algorithms incorporating branching and iterations, and test, modify and implement digital solutions. They evaluate information systems and their solutions in terms of meeting needs, innovation and sustainability. They analyse and evaluate data from a range of sources to model and create solutions. They use appropriate protocols when communicating and collaborating online.		
Strands	Digital Technologies processes and production skills • Collecting, managing and analysing data • Creating digital solutions by: – Investigating and defining – Producing and implementing		
Content descriptions	• Analyse and visualise data using a range of software to create information, and use structured data to model objects or events (ACTDIP026) • Define and decompose real-world problems taking into account functional requirements and economic, environmental, social, technical and usability constraints (ACTDIP027) • Implement and modify programs with user interfaces involving branching, iteration and functions in a general-purpose programming language (ACTDIP030)		
Key concepts	• data collection • data interpretation • specification • algorithms • implementation	***Key ideas***	Thinking in Technologies • computational thinking
Cross-curriculum priorities		***General capabilities***	• Information and Communication Technology (ICT) Capability • Literacy • Numeracy

Useful links

- Find out more about the micro:bit at www.microbit.org
- Code the micro:bit at www.makecode.org

Links to Mathematics

To explore a purposeful connection to Mathematics, students should discuss possible outcomes and calculate the expected probabilities before attempting to program or collect data from the micro:bit. The main activity described in this resource is focused on data collection and representation. Students are collecting data and representing the outcomes graphically. Consider including in the lesson plan that students will identify the expected outcomes and probabilities before choosing the code. They can then compare their predictions with the results they collected during the micro:bit tasks. Students could also compare the coin toss and dice roll as random experiments and compare them to their theoretical probability (ACMSP146). Year 6 students could also investigate recording probability as percentages and decimal representations, which forms part of learning in the 'Number and Algebra' strand. Refer to: https://www.australiancurriculum.edu.au/f-10-curriculum/mathematics/

Mathematics Year 6

Content strand: Statistics and probability
Sub-strand: Chance
Content description:

- Describe probabilities using fractions, decimals and percentages (ACMSP144)

Elaboration:

- investigating games of chance popular in different cultures and evaluating the relative benefits to the organisers and participants (for example Pachinko)

Content description:

- Conduct chance experiments with both small and large numbers of trials using appropriate digital technologies (ACMSP145)

Elaboration:

- conducting repeated trials of chance experiments, identifying the variation between trials and realising that the results tend to the prediction with larger numbers of trials

Content description:

- Compare observed frequencies across experiments with expected frequencies (ACMSP146)

Elaboration:

- predicting likely outcomes from a run of chance events and distinguishing these from surprising results

Mathematics Year 7

Content strand: Statistics and probability
Sub-strand: Chance Content description:

- Construct sample spaces for single-step experiments with equally likely outcomes (ACMSP167)

Elaboration:

- distinguishing between equally likely outcomes and outcomes that are not equally likely

Content description:

- Assign probabilities to the outcomes of events and determine probabilities for events (ACMSP168)

Elaboration:

- expressing probabilities as decimals, fractions and percentages

Mathematics Year 8

Content strand: Statistics and probability
Sub-strand: Chance
Content description:

- Identify complementary events and use the sum of probabilities to solve problems (ACMSP204)

Elaboration:

- understanding that probabilities range between 0 to 1 and that calculating the probability of an event allows the probability of its complement to be found

Note: ACMSP204 would work well as heads and tails are complementary events; rolling a 5 and not rolling a 5 are complementary but rolling a 5 or rolling a 6 are not.