Curriculum coverage

This chapter covers these requirements for the Computing Programme of Study (age 11–14):
- design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems
- understand several key algorithms that reflect computational thinking.

This chapter also covers these main requirements for the Computing at School (CAS) Progression Pathways (for a full list of requirements met, see pages 8—9 of this guide):
- understand that iteration is the repetition of a process such as a loop
- recognise where information can be filtered out in generalising problem solutions
- represent solutions using a structured notation.

Preparation

Computational thinking is a way to study computing and approach problem solving in everyday life. Computational thinking helps students think through problems, break problems down into smaller parts and find good solutions. The foundations of computational thinking lie in logical reasoning. Logical reasoning is useful in many subjects, from science to history.

If you are teaching this topic for the first time, you may want to work through the student exercises before each lesson. You could also develop your own teaching examples to show students how they can apply computational thinking concepts to daily life in their own local context.

Games may be useful to help students learn the technical key words in this chapter. Create several sets of cards with individual key words on them. You can take these words from the Key words boxes in each lesson of the Student Book. Use all the terms for the year and play different word games as the year progresses. Only use words the students understand for each lesson. Some suggestions for games are given in this Teacher’s Guide.

Learning outcomes

In this chapter students use computational thinking to solve the problem of designing a simple spam filter. First, students will learn to decompose the problem. Then they will find out how to recognise patterns when solving computing problems. Students will also discover how to represent problems using flowcharts and evaluate their solutions.

By completing this chapter students will be able to:
- describe computational thinking
- use decomposition to break a problem down into smaller parts
- describe an algorithm
- use pattern recognition
- use a flow chart to describe their problem solving
- describe selection
- use if... then... else to navigate through a problem
- describe iteration
- use loops to navigate through a problem
- evaluate other people’s work
- give helpful feedback.
**Design a spam filter**

Many of the activities in this chapter do not require a computer. Computational thinking is a way of thinking and solving problems. It can be represented with or without technology.

**Offline activity**

The introductory pages in the Student Book introduce the idea of computational thinking to students. Here is an extra offline activity that you can use to introduce computational thinking. You can use this task if the school computers are down, or if you need to work in a room with no computers. You will need: blindfolds, pieces of paper with one large coloured dot on each piece. Ask students to work in pairs or small groups. Give each group a colour. In each group, ask one student to cover his or her eyes or wear a blindfold. Now place the coloured dots around the room. Each group must give verbal instructions to the blindfolded student. The blindfolded student must find the coloured dot for their group. Ensure that obstacles such as chairs and bags are tidied away before you start.

Before you play the game ask students to think about:

- the problems they might face, for example, each group shouting instructions at the same time might be confusing for the blindfolded student
- ways of solving the problems
- the kinds of instructions that might be useful.

**Talk about...**

The discussion is also an activity you can do offline. You could use this activity any time to vary the pace of lessons and encourage students to reflect on their learning.

Discuss the blindfold game you have played. Support the students in stating the problems they faced and the solutions they found.

Explain that computational thinking is a way of understanding and solving problems that can help us in computing and in our daily lives.

Ask students to discuss problems they encounter in their everyday lives. The Student Book gives two examples, but other examples you could discuss are:

- planning a meal
- preparing your clothes for a physical education lesson
- keeping in touch with family who live far away.

Choose a problem that interests students. Break the problem down into smaller parts. For example, the problem of keeping in touch with faraway family could include:

- whether you have access to a computer, or phone, and the internet
- whether your faraway family have access to a computer or phone and the internet
- whether there is a time difference between the two locations.

**FACT**

Seymour Papert was a computer scientist, mathematician and teacher. He was born in South Africa in 1928 and has lived and taught all over the world. His book *Mindstorms: Children, Computers and Powerful Ideas*, tells us that the ways children learn to use technology can make a difference to the ways they learn everything else.

http://www.papert.org/

**Word cloud**

The word cloud contains all the key words that have been highlighted and defined in Key words boxes throughout the lesson. The key words for this chapter are: process, computational thinking, spam filter, spam, algorithm, implement, loop, decomposition, flow chart, pattern recognition, sequencing, selection, iteration, variable, criteria, fit for purpose, efficient, elegant, data, command, sequence.
1.1 What is computational thinking?

Learning outcomes
When they have completed this lesson students should be able to:
- describe computational thinking
- use decomposition to break a problem down into smaller parts.
More confident students will:
- begin to identify and explain patterns.

Overview
This is the first lesson introducing computational thinking. Students do not need any prior learning in the subject. In this lesson students will find out more about what computational thinking is. Students learn why computational thinking is useful in computing and in everyday life. Students also apply computational thinking to the problem of designing a spam filter as their project for this chapter.

Language development
Students will discover that computational thinking is a way of thinking through problems and finding effective solutions. Students are also introduced to the term algorithm, which is a set of instructions or rules that we can follow to carry out a task. The word ‘decomposition’ has a technical meaning in computational thinking. Decomposition means breaking down a problem into smaller parts. Based on your knowledge of students’ language competencies, you may want to compare this meaning with a science-based meaning. For example, decomposition also refers to the process of organic matter decomposing in a composter. Students are also introduced to the word ‘spam’. Understanding that some emails are unsolicited and unwelcome is an important part of helping students keep safe online.

Before the lesson
You may want to set up an email account for the whole class. You could use this account to monitor incoming spam emails and discuss these, where appropriate, with students. Monitoring incoming spam emails would ensure that the students’ base their understanding on real-life examples.

The key words for this lesson are: algorithm, computational thinking, decomposition, process, spam and spam filter. The words are highlighted when they first appear in the text. Their definitions are included in the Key words box at the end of the lesson. You may want to review these words before the lesson.

Learn about...
You will lead the first part of the lesson. Make sure students understand these ideas. You may ask them to make notes. You may use directed questioning to check understanding.
- **Algorithm**: An algorithm is a set of instructions or rules that we make to carry out a task or solve a problem.
- **Decomposition**: We use decomposition to break the problem down into smaller parts that are easier to solve.

How to...
In the second part of the lesson, students complete an exercise under your guidance.

Look at the three smaller parts of the decomposed problem:
- What type of chocolate would sell well?
- How would you make it?
- How would you get it to customers?

Ask students to talk about whether the first decomposed problem is complete. The factory owners might also want to think about:
- **Price**: Is it important to think about how much money the factory would like to make first? How would this make a difference to the rest of the process?
- **What ingredients can the factory use?**
- **Where will the chocolates be selling?** What do people in those places like to eat?

Tell students that there are no right or wrong answers because context makes a big difference to any decomposition process. Decomposing a problem
correctly means we need to think about the important factors in any particular context. Read through the algorithms with students. Ask whether they agree with the algorithm given. Look at the task ‘Find out what type of chocolate customers would like to buy’. The first line of the algorithm is ‘Invent some chocolate ideas’. Ask students if they think they could decompose the task further. They might suggest:
- bringing together a group of children to invent a new idea
- carrying out market research in a shop to find out what people would like.

Now you do it...

Students apply decomposition and algorithmic skills to the main project for this chapter, which is to design a simple email spam filter. Read through the text of this learning activity with students so they understand the idea of spam. Ask whether any of them have seen a spam email in real life. Explain that there are two ways to protect themselves from spam. They can:

1. Learn to recognise spam in their email inboxes, and delete or report it if they see it.
2. Use an effective spam filter.

Students will work in pairs, using the two spam examples to answer the questions.

What success looks like:
- Who is the email from? Answer: Example 1: Mrs Smith; Example 2: Honest Bank.
- Does the name on the email match the email address? Answer: Example 1: No; Example 2: Yes and no. We would expect the email address to be from honestbank.com, not cqhy457.com.
- Is the content of the email addressed to the email owner? Answer: Both examples: No.
- Does the email ask for personal details? Answer: Example 1: No; Example 2: Yes
- Does the email ask for money? Answer: Example 1: Yes; Example 2: No.
- What are the key words that your spam filter will detect? This is a more challenging question, and students may need you to explain the meaning of ‘key word’ and ‘detect’. Answer: A range of answers is possible, including: checking, account, bank, banking, help, trouble, stranded, bill, cheque, $, temporary, post.

If you have time...

Working with a partner, students make a list of four items. Three items on the list should have things in common and one should be the odd one out. Ensure the students understand the rules of the game. Explain that there are no right or wrong answers. The game will help more able students begin to identify patterns.

What success looks like: The students can correctly identify the odd one out.

Test yourself...

FOUNDATION QUESTIONS
1. What is an algorithm? Answer: An algorithm is a set of instructions or rules we can use to carry out a task. Students should include the words ‘instructions’ or ‘rules’ in their answer.
2. Decompose the problem of getting dressed in the morning. Answer: A range of answers is possible, but students should identify that they will need to put on their underwear before outerwear, day clothes before coats, socks before shoes, etc.

EXTENSION QUESTIONS
3. What should you do if you receive a spam email? Answer: Learn to recognise spam in your email inboxes, and to delete or report it. Use an effective spam filter.
4. Give an example of how computational thinking relevant to everyday life. Answer: Students’ own example including decomposition of a problem.
1.2 Pattern recognition

Learning outcomes

When they have completed this lesson students should be able to:

✶ use pattern recognition.

More confident students will:

✶ be able to compare two problems.

Overview

In this lesson students will learn pattern recognition. Being able to identify and create patterns is an important tool in their computational thinking toolset. Students will:

✶ discover the importance of pattern recognition in solving real-world problems
✶ carry out a simple pattern recognition task as a class
✶ apply these new skills to the spam filter they are designing as the chapter project.

Language development

This lesson asks students to apply their knowledge of English to identify common word patterns in spam emails.

Before the lesson

You might want to share an enlarged version of the spam emails to help your classroom discussions.

The key words for this lesson are: pattern recognition. The words are highlighted in the text the first time they appear. Their definitions are included in the Key words box at the end of the lesson. You may want to review these words before the lesson.

Learn about...

You will lead the first part of the lesson. Make sure students understand these ideas. You may ask them to make notes. You may use directed questioning to check understanding.

✶ What is a pattern? A pattern is finding the similarities and differences between things. Look at the example of the group of people in the Student Book. Ask students to suggest patterns in everyday life, such as patterns in nature (e.g. trees) or in equipment (e.g. balls for playing sports).

✶ We need to identify patterns when we solve real-world problems. Look at the example of Ignaz Semmelweiss in the Student Book.

✶ We can instruct computers to recognise patterns for us. Computational thinking uses pattern recognition to help solve problems.

How to...

In the second part of the lesson, students complete an exercise under your guidance.

✶ Work with students to identify common words and features in the two mock spam emails from Lesson 1.1. The Student Book gives examples, such as ‘bargain’ and ‘money’.

✶ Ask students why these words and features might seem suspicious. Students might suggest that unknown people are referring to money and that the language is too friendly.

✶ Discuss whether a bank is likely to send emails to their clients.

Now you do it...

In this lesson, students are asked to refer back to their answers from the learning activity in Lesson 1.1. These answers form the basis of a decomposed problem. Students work in pairs to answer the questions in this activity.

What success looks like:

✶ Can you identify any patterns that might be useful? Answer: Both emails discuss money; neither email is addressed personally; individual words such as ‘money’ and individual signs such as ‘$’ are used.

✶ What parts of the email could you think of as patterns? Answer: The sender’s email address, greeting, body of the email can be seen as patterns.

If you have time...

Ask more able students to write algorithms for making a cup of tea and a cup of coffee.
What success looks like:

- Which tasks are similar? Answer: Walk to the cupboard; take out a cup; place the cup on the table; pick up the kettle; take the kettle to the tap; fill the kettle with water; put the kettle on the table; switch on the kettle; pour boiled water into the cup.
- Which tasks are different? Answer: Take out the instant coffee; measure coffee into the cup; take out the teabag.
- Where could you use the same instructions in both processes? Answer: Walk to the cupboard; take out a cup; place the cup on the table; pick up the kettle; take the kettle to the tap; fill the kettle with water; put the kettle on the table, switch on the kettle; pour boiled water into the cup.

Test yourself...

FOUNDATION QUESTIONS

1. What is pattern recognition? Answer: Pattern recognition is finding things that decomposed problems have in common.

2. Here is an image of some robots. Write any patterns you see. Answer: A range of answers is possible. Students should be expected to note similarities and differences. Similarities: All the robots have at least one eye; all the robots have a device for standing; all the robots can move; all the robots have something carrying out the function of a mouth. Differences: Not all of the robots have arms; each robot is a different colour; each robot is made of different shapes.

EXTENSION QUESTIONS

In Lesson 1.1, you decomposed the problem of designing and selling a new type of chocolate. Now imagine you are also going to design and sell a new type of cake.

3. Decompose the cake problem. Answer: A range of answers is possible. Many students may want to refer to the relevant text in Lesson 1.1 to provide a structure for their decomposition. The decomposition process can be repeated by replacing ‘chocolate’ with ‘cake’.

4. Write down the patterns you can see between the chocolate design process and the cake design process. Answer: Students should notice that the processes are almost identical.

Overview

In this lesson students will be introduced to flow charts. Flow charts are an important way of representing computational thinking. We need to be able to represent our thinking to communicate it to others. Representing our thinking also helps make sure we have decomposed the problem properly. Students will learn about the basic components of flow charts. Students will also begin to describe their computational thinking on the spam filter problem by using a flow chart.

Language development

This lesson guide offers an additional lesson activity at the end, which will help the class develop a glossary to support language development as the chapter progresses.
Before the lesson
The key words for this lesson are: command, data flow chart, sequence and variable. The words are highlighted in the text the first time they appear. Their definitions are included in the Key words box at the end of the lesson. You may want to review these words before the lesson.

Learn about...
You will lead the first part of the lesson. Make sure students understand these ideas. You may ask them to make notes. You may use directed questioning to check understanding.

- An algorithm is a set of instructions or rules we can use to carry out a task.
- Planning an algorithm carefully is likely to lead to an algorithm that is correct and efficient.

So far in this chapter, students have written their algorithms as a long list of instructions. In the real world, people use flow charts and pseudocode to communicate their ideas. People use flow charts and pseudocode to test their algorithms before they implement these algorithms in computer code. (Students will learn more about pseudocode in Matrix 2.) Many people find flow charts useful because they are a way of communicating ideas. Flow charts promote accuracy and make it easier to spot mistakes.

How to...
In the second part of the lesson, students complete an exercise under your guidance.

First, students are introduced to the most commonly-used flow chart symbols. You may want to spend time on the word ‘variable’, as this is an important concept in computing and science. The Khan Academy website has some helpful analogies on variables: www.khanacademy.org

Work through the flow chart example with students.

Now you do it...
Students make a flow chart to describe the program for their spam filter. Gaps in students’ knowledge will become clear during this task. Before they complete the task, explain that they will be working on improving their flow chart for the remainder of this chapter. Students should not expect to produce a final or ‘correct’ version in this lesson. This lesson is an opportunity for students to learn that computing often requires trial and error ways of working.

Working in pairs, students can work through the implications of the process-decision outlined in the Student Book. Ask students to think about what happens just before and just after these steps. Some students may prefer to begin with the ‘start’ command and progress forward.

What success looks like: Students begin to draft a flow chart for their spam filter.

Keep in your mind how the final flow charts might look to support your progress through the chapter. You can see this in Lesson 1.4 of this guide.

End the lesson with a game that reinforces students’ learning of new words. Put students into groups. Give each group a set of the key words cards you made earlier. Ask each group to create as many definitions for the words on the cards as they can in a five-minute period. Once the five minutes are over, ask each group to read out their definitions. Students could then create a classroom glossary to hang on the wall, which will support their language development.

If you have time...
Students are asked to think of any disadvantages of using flow charts to show their computational thinking.

What success looks like:

- If a change has to be made, the flow chart would need to be drawn again.
- Complicated processes can be difficult to represent.
- Flow charts can be slow to create.
- Some people find flow charts forced—flow charts do not suit the way they approach computing problems.

Test yourself...

FOUNDATION QUESTIONS
1. Why are flow charts useful in computational thinking? Answer: Flow charts are a way of communicating ideas to others; they make it easier to spot where there might be mistakes in a code; flow charts promote accuracy.

2. Draw and label the four main shapes we use to make flow charts. Answer: Students should draw the Start/Stop, Process, Decision and Input/Output shapes on page 17 of the Student Book.

EXTENSION QUESTIONS
3. Decompose the task of brushing your teeth. Answer: A range of answers is possible. Students should list many of these tasks: Pick up toothbrush; rinse toothbrush with water, pick up toothpaste, open toothpaste; squeeze toothpaste onto toothbrush; move toothbrush around mouth in circular motion; put down toothbrush; pick up glass
of water; rinse mouth with water; spit out water; rinse toothbrush.

4 Make a flow chart to describe the process of brushing your teeth. Answer: Students’ flowcharts should include Start/Stop shapes and a series of processes. The most able students could include a decision point when deciding whether to stop brushing teeth.

1.4 Selection and if... then... else statements

Learning outcomes
When they have completed this lesson students should be able to:
- describe what selection is
- use if... then... else to navigate through a problem.

More confident students will:
- have applied their understanding of selection to real-life computer program examples.

Overview
In this lesson students will be introduced to two important tools in designing algorithms: sequencing and selection. Students will learn when it is useful to use these tools and how to represent them in a flow chart. The lesson encourages students to think about the decision points in their spam filter flow charts. Students will also think about choosing sequencing or selection tools.

Language development
Students are introduced to three key words in this lesson. They are all words that are used more widely in English, but have technical meanings in computing. Sequencing is the order in which steps or tasks are carried out. Selection happens when a step in the algorithm is reached where the program has two or more possibilities of what to do next. Implement means to put an algorithm into action. You may want to spend time comparing the ways in which these words are used in English and the ways in which they are used in computing.

Before the lesson
The key words for this lesson are: implement, selection and sequencing. The words are highlighted the first time they appear in the text. Their definitions are included in the Key words box at the end of the lesson. You may want to review these words before the lesson.

The next section of this guide describes a dominoes game that you may want to play in the first part of the lesson ‘Learn about...’. If you want to play the dominoes game, you will need dominoes, or rectangular prism wooden blocks of identical size. If not, you may want to share a suitable YouTube video of a domino run.

Learn about...
You will lead the first part of the lesson. Start with a game. Put the students into groups. Give each group up to twenty dominoes or blocks. Ask each group to set up a ‘domino run’ by placing each block on its smallest end, in a line, with about one centimetre between the blocks. Ask the students to set off the domino run by pushing the block on one end into the block next to it.

Explain that the domino run is like a sequence. Each block carries out its own task in a particular order.

Now ask the students to create a run with a single line that then branches out into two lines.

Explain that this is like selection. The algorithm has a decision point with two (or more) possibilities of what to do next.

Make sure students understand these ideas. You may ask them to make notes. You may use directed questioning to check understanding.
- **Sequence**: A sequence is the order in which steps or tasks are carried out.
- **Selection**: Selection happens when two or more things could occur in a step in a program. We can talk about selection as a decision, for example a computer avatar could decide to walk down a path. We can talk about selection as a question, for example by asking whether a robot has done an activity enough times.

**How to...**

In the second part of the lesson, students complete an exercise under your guidance.

This section in the Student Book explains how to implement selection in an algorithm. Building on the example from Lesson 1.3, it introduces the students to three important ideas:

- We can use ‘yes’ and ‘no’ to help our algorithm decide which path to take. If the answer to the decision point is yes, the algorithm can be instructed to take a particular path. If the answer to the decision point is no, the algorithm can be instructed to take a different path.
- We can use `if… then` to say this in a more elegant way. If the answer to our decision is yes, then the algorithm can be instructed to take a particular path. If the answer to the decision is no, then the algorithm can be instructed to take a different path.
- If there are more than two choices we may want to use `if… then… else`.

Students will build on these ideas during the programming chapters (Chapter 2, App Inventor and Chapter 4, Introducing Python) in the Student Book.

In this lesson, students apply what they know about selection to their developing flow charts. Here is an example of what a simple spam filter flow chart might look like by the end of the chapter. Students are not expected to reach this stage in this lesson.

**Now you do it...**

In this activity, students work with a partner to apply what they know about selection to their flow chart showing the design of a spam filter.

**What success looks like**: When they reach their decision points, they should think of the possible
answers to the question. They should show the
different paths their design could take. Students’
answers might include: a computer game where an
avatar has a choice of more than one route to take; a
logistics program choosing routes for delivery
vehicles or ambulances; an online survey.

If you have time...
Students are asked to think about other computer
programs where selection might be useful.
**What success looks like:** A range of answers is
possible, including: a computer game where an
avatar has a choice of more than one route to take; a
logistics program choosing routes for delivery
vehicles or ambulances; an online survey.

Test yourself...
**FOUNDATION QUESTIONS**
1 Finish this sentence: Selection happens when...
Answer: …a step in the algorithm is reached where
the program has two or more possibilities of what
to do next.
2 What does implement mean? Answer: Implement
means to put algorithms into action.

**EXTENSION QUESTIONS**
Work with a partner. Imagine you are programming a
game. Your avatar is walking along a woodland path.
The avatar sees three paths ahead.
3 Decompose the problem. Are there obstacles on
any of the paths? Is there treasure on any of the
paths? Answer: A range of answers is possible.
Expect students to describe at least one obstacle
or reward for the three paths.
4 Create a simple flow chart that shows what they see
when looking down each one of the paths. Answer:
A range of answers is possible, depending on the
obstacles or rewards the students have described in
Question 3. Expect students to use selection
constructs. Most students will use
if … then.
More able students will use
if … then … else.

---

1.5 Loops

**Learning outcomes**
When they have completed this lesson students should be able to:
- describe iteration
- use loops in a program.
More confident students will:
- extend their understanding of loops beyond the lesson.

**Overview**
In this lesson the students will learn about the
construct of iteration, and how to use loops in
computing to show iteration. Students will apply
their understanding to their spam filter designs. By
the end of the lesson, students should be close to
completing their flow charts. Students will evaluate
these flow charts in the final lesson of the chapter.

**Language development**
This lesson introduces students to the idea of
iteration and the concept ‘to loop’. These words are
used widely in English, but have technical meanings
in computing. Iteration means repeating a step or
task in an algorithm or computer program. To loop
means to go back to a previous step. Looping
happens in iterations.
Before the lesson

You will need to prepare pieces of string before the lesson to carry out the language reinforcement activity. Before the lesson evaluate each student’s progress against the chapter objectives. During the lesson work with any students who have not yet created a flow chart for a spam filter. Working with these students will ensure they have something that can be evaluated during the final lesson.

The key words for this lesson are: iteration and loop. The words are highlighted the first time they appear in the text. Their definitions are included in the Key words box at the end of the lesson. You may want to review these words before the lesson.

Learn about...

You will lead the first part of the lesson. Make sure students understand these ideas. You may ask them to make notes. You may use directed questioning to check understanding.

- **Constructs**: Constructs are the tools we use to build algorithms. The students already know about two constructs: sequencing and selection.
- **Iteration**: Iteration is another construct. Iteration makes it possible for us to tell an algorithm to repeat a step more than once. We show iteration in a flow chart using loops. We can show a loop in a flow chart using an arrow.

If students need more reinforcement of the language, give each student a piece of string. Ask them to place all the pieces of string end-to-end. The pieces of string are now in a sequence. Now ask each student to make a loop by joining the ends of their own piece of string. Place the loops next to each other. Explain to students that the loops are iterating (repeating).

How to...

In the second part of the lesson, students complete an exercise under your guidance. The exercise is based on a computer game, in which an avatar looks down three paths in a wood. Each path contains treasure. The avatar is instructed to walk down each path, collect the treasure, and return to the start.

Move through the flow chart example step by step (page 25).

- Start.
- There are three paths ahead of the avatar.

Now you do it...

Students work in pairs to adapt their flow chart to include a loop. Students should see how the loop makes repeated actions, such as searching for key words. The completed flow chart in this Teacher’s Guide (Lesson 1.4) shows how students could respond to the questions in the Student Book.

**What success looks like**: Students’ answers might include:

- There are repeated actions searching for multiple key words in each email. There are also repeated actions in searching more than one email.
- Loops are an efficient way to carry out repeated actions without having to write the same instruction many times.

Encourage students to redraw their flow chart so it can be clearly read by others for the next lesson. Work with less able students to complete a flow chart together.

If you have time...

This extension activity refers to the ‘How to’ section for this lesson and the test question in Lesson 1.4. The extension activity asks students to think about what changes they might make to the flow chart if there were more than one path.

**What success looks like**: More able students should realise that they would not need to change their flow chart for ten paths, or for a million. This is why abstraction, iteration and looping are useful in computing, and why they are essential computational thinking skills.
Test yourself...

FOUNDATION QUESTIONS
1. Complete the words naming the three algorithm constructs we use in computational thinking. Answer: Sequencing; Selection; Iteration.

2. Why is iteration important in programming? Answer: Iteration makes it possible to repeat a command or action many times without needing to write or code it many times.

EXTENSION QUESTIONS
3. Think of an example of a computer program that would need to use loops. Answer: Students could suggest a range of programs, including software that measures and tracks the weather, stock markets and computer gaming.

4. Using your own words, describe what is happening in this flow chart. Answer: Students are told that the flow chart describes the actions of the factory robot. Their responses should contain these points:
   - Robot aligns with Point A.
   - At the decision point, ask if there is a box for the robot to pick up. If the answer is NO, then stop. Some less able students might say ‘if there is no box, then stop’. If the answer is YES, then pick up the box. Some less able students might say ‘if there is a box, pick it up’.
   - Move the box to Point B.
   - Put down the box.
   - Move back to Point A. Loop back to the alignment step.

Overview
This final lesson gives students the opportunity to complete their work, evaluate others’ work and learn to provide effective feedback. You may want to use this lesson to complete formative assessments and give small group or one-to-one support.

Language development
Students will learn to use everyday English words to describe the evaluation process. However, the word ‘elegant’ may be confusing for students with English as an additional language. Elegant has a different meaning in computing to the one it has in everyday English. Elegant means that an algorithm is clear for someone else to understand. You can discuss the meaning of this word during the lesson.

Before the lesson
Be aware which students will need to use this lesson to complete their spam filter design, and which students are likely to complete the extension task. The key words for this lesson are: criteria, efficient, elegant and fit for purpose. The words are highlighted the first time they appear in the

Learning outcomes
When they have completed this lesson students should be able to:
- evaluate other people’s work
- give helpful feedback.
More confident students will:
- have applied their feedback to their work.

1.6 Evaluation
text. Their definitions are included in the Key words box at the end of the lesson. You may want to review these words before the lesson, particularly the word, ‘elegant’ as this has a different meaning in computing.

If you think students will find the words used during this lesson difficult, make sets of cards from the set of key word cards for this lesson (see the Preparation section) and create a matching set of definition cards. Have several copies of each set of words for each pack of cards you make. Each pack will be used by two students, playing in pairs, for the snap game suggested in the first part of the lesson.

Learn about...

You will lead the first part of the lesson. Make sure students understand these ideas. You may ask them to make notes. You may use directed questioning to check understanding.

- Once we have designed an algorithm, we can evaluate it to see if it is fit for purpose.
- An algorithm can be hard to evaluate when we have been working closely on it. To solve this problem we can ask other people to evaluate our work, using agreed criteria.
- An algorithm that is fit for purpose is elegant, efficient, decomposed and correct.
- Elegant means that an algorithm is clear for someone else to understand. Efficient means that an algorithm is doing the best work for the least effort. Fit for purpose means that an algorithm or program does the job it is supposed to do. We use criteria to judge whether a program is doing the job we need it to do.

If students find these words hard, play a snap game. Place the students in pairs. Students take turns to put cards on the table. When the word matches the definition, the student identifying the match calls out ‘snap’. The student can then pick up all the cards on the table. The student with the most cards wins the game. This game will help students who are finding the language difficult to practice their understanding in a fun way.

How to...

In the second part of the lesson, students complete an exercise under your guidance. The example is a simple algorithm for making a cup of tea.

Discuss the first algorithm with students. Use the key words for the lesson to describe why the algorithm is not fit for purpose.

- Look at the properly decomposed algorithm beneath the tea pot diagram. Compare and contrast the differences between the algorithms with students. Point out that although the second algorithm is longer, it addresses the criteria more completely than the short algorithm. The short algorithm was not fit for purpose.
- Explain how to give effective feedback.
  Encourage the students to use sentence openings such as:
  - “You could improve this by…”
  - “I like the way you have…”
  - “If you added….this algorithm would be more elegant/fit-for-purpose/efficient/correct.”
- You may want to give some silly examples to show students which behaviours and language to avoid. For example:
  - “This is good.” (unclear)
  - “I think you have done this wrong.” (personal and subjective)
  - Any statement followed by ‘but’.

Now you do it...

Work as a class to develop three evaluation criteria for the spam filter algorithms that students have completed.

Still working in pairs, students who have completed their algorithms can evaluate each others’ work, giving effective feedback. Ask students to write down their feedback so you can assess it after the lesson. While students write their feedback, you can offer small group or one-to-one support to those who have not finished their algorithms.

What success looks like: Students complete their algorithms and evaluate each others’ work, giving effective feedback.
If you have time...
The Student Book asks more able students to implement the feedback they have been given. If students do not agree with the feedback, encourage them to say why they do not agree.

What success looks like: Students implement their feedback and can discuss it, giving reasons for their answers.

Test yourself...

FOUNDATION QUESTIONS

1. What criteria can we use to judge whether an algorithm is fit for purpose? Answer: decomposed, correct, efficient, elegant.

2. What are the characteristics of good feedback? Answer: Keep it about the work; show where things are working well; be specific.

EXTENSION QUESTIONS

3. Summarise the main strengths and weaknesses of the spam filter you have designed. Answer: Answers will depend on the characteristics of the individual algorithms students have designed. Students’ responses to this question should take into account the four characteristics of a fit-for-purpose algorithm outlined in this lesson: decomposition, correctness, elegance, efficiency.

4. How could you address the weaknesses? Answer: Answers will depend upon the characteristics of the individual algorithms students have designed. Students’ responses should demonstrate self-awareness and the ability to be critical about their own work.

Review what you have learned about computational thinking

The test questions and assessment activities in the Student Book give you an opportunity to evaluate students’ understanding. The questions are shown here with possible answers.

Model answers to test questions

1. What is computational thinking? Answer: Computational thinking is making sure we understand problems and thinking up possible solutions.

2. Why is computational thinking important to programming? Answer: Computational thinking is important because it helps us: understand problems, find the right solutions to problems, communicate problems and solutions to others, solve problems together.

3. Give an example of how we can use computational thinking in everyday life. Answer: We might use computational thinking to: plan a trip, plan a written piece of work, make a meal, create a piece of artwork, set up a business.

4. What are the three constructs we can use to design algorithms? Answer: a) iteration, b) selection, c) sequencing.

5. Explain what the following words and phrases mean using your own words. Answer: a) pattern recognition: finding the similarities and differences between things; b) sequencing: the order in which steps or tasks are carried out; c) selection: happens when you reach a step in the algorithm where the program has two or more possibilities of what to do next; d) iteration: repeating a step or task in an algorithm or computer program.
6 What is a loop? Answer: Most students’ answers to this question should include an explanation that looping means returning to a previous step. More able students may place this in the context of iteration.

7 Give an example of an algorithm where a loop would be useful. Answer: A loop would be useful: when searching for a key word in a long text or when a robot needs to repeat the same action. For example, an underwater robotic vehicle might need to move back and forth across the ocean floor when looking for something.

8 Think back to the algorithm evaluation jigsaw. Which four words can you use to help us decide whether an algorithm is fit for purpose? On a separate sheet of paper, draw the jigsaw and write the letters ‘a’ to ‘d’ on each piece. Answer: a) decomposed, b) efficient, c) correct, d) elegant.

9 Design a poster to tell other students about the dangers of spam. Include this information on your poster:

   a) What is spam? Answer: Email spam involves sending messages to many people who have not asked for the emails to be sent.

   b) What is a spam filter? Answer: A spam filter is a computer program that spots unsolicited email. It can stop unsolicited email from getting into your inbox.

   c) How does a spam filter work? Answer: A spam filter searches for key words and phrases in every email sent to your inbox. A spam filter identifies emails that are likely to be untrustworthy.

   d) What should you do if you receive a spam email? Answer: Most students should give at least one of these answers. More able students may give both answers.

      o Delete or report spam email.
      o Use an effective spam filter.

10 Look at the poster another student has made. Evaluate the poster by telling the other student one thing that has been done well, and one thing that the student could do to improve the poster. Answer: Student feedback on the poster could be written or spoken, depending on your knowledge of the class. The feedback should:

      o be specific
      o be focused on the work and not include any personal negative comments
      o show where things have been done well.

✔ Model answers to assessment activities

Starting activity

Students are given a choice of three meals to make: a salad, a boiled egg or a sandwich. The students are asked to choose one meal and decompose the task.

What success looks like:

Making a sandwich is the easiest of the three tasks. The task does not specify a sandwich filling or type of bread. Answer: Expect to see some or all of these decomposed tasks: Place bread on breadboard; Fetch knife; Cut two slices of bread; Fetch butter (or other choice of filling); Spread butter on bread; Place one piece of bread on the other.

Making a salad is a more challenging task, as students will need to include a loop in their flowchart later in the assessment. Answer: Expect to see some or all of these decomposed tasks: Choose type of salad; Fetch ingredients from the fridge, place on table; Fetch chopping board and knife, place on table; Fetch a bowl, place on table; Chop vegetables; Place vegetables in bowl.

Expect more able students to break the task down further if necessary. For example, they might include more detail on ingredients, or include ingredients that require cooking, e.g. rice.

Boiling an egg is a more challenging task. More able students may choose to include decision points and loops at up to two points in their flow charts later in the assessment. Answer: Expect to see some or all of these decomposed tasks: Take pot out of cupboard; Pour water into pot; Place top on stove; Boil water; Fetch egg; Place egg in pot of water; Record time; Check time; If correct time has elapsed, remove egg. If not, go back to check time; Place egg in egg cup or remove egg shell.
Intermediate activity
Students are asked to draw a flow chart to show how you would make their chosen meal.

What success looks like: All students should include most or all of these elements in their flow charts.

Extension activity
Students are asked to evaluate the flow chart made by a partner.

What success looks like: The written feedback they provide should include the characteristics of helpful feedback. They should:
- be specific
- be focused on the work, and not include any personal negative comments
- show where things have been done well.

Matrix Teacher Guide 1

Making a sandwich

Start
Place bread on breadboard
Fetch knife
Cut 2 slices of bread
Fetch butter and/or filling
Spread butter and/or add filling to bread
Place 1 piece of bread on the other
Stop

Making a salad

Start
Choose type of salad
Fetch ingredients from fridge, place on table
Fetch chopping board and knife, place on table
Fetch a bowl, place on table
Chop ingredient 1
Is there enough of ingredient 1?
Yes
Chop ingredient 2
Is there enough of ingredient 2?
Yes
Place vegetables in bowl
Stop

Boiling an egg

Start
Place bread on breadboard
Fetch knife
Cut 2 slices of bread
Fetch butter and/or filling
Spread butter and/or add filling to bread
Place 1 piece of bread on the other
Stop

Start
Take pot out of cupboard
Pour water into pot
Place pot on stove
Boil water
Place egg in pot
Record time
Check time
Has correct time passed?
Yes
No
Is there enough of ingredient 1?
Yes
Stop

Stop

Stop