Computational and
algorithmic thinking
in Mathematics

Unpacking the content descriptions

Level 10



Authorised and published by the Victorian Curriculum and Assessment Authority
Level 7, 2 Lonsdale Street
Melbourne VIC 3000

© Victorian Curriculum and Assessment Authority 2019

No part of this publication may be reproduced except as specified under the *Copyright Act 1968* or by permission from the VCAA. Excepting third-party elements, schools may use this resource in accordance with the [VCAA educational allowance](https://www.vcaa.vic.edu.au/Footer/Pages/Copyright.aspx#schools). For more information go to: <https://www.vcaa.vic.edu.au/Footer/Pages/Copyright.aspx>.

The VCAA provides the only official, up-to-date versions of VCAA publications. Details of updates can be found on the VCAA website: [www.vcaa.vic.edu.au](http://www.vcaa.vic.edu.au).

This publication may contain copyright material belonging to a third party. Every effort has been made to contact all copyright owners. If you believe that material in this publication is an infringement of your copyright, please email the Copyright Officer: vcaa.copyright@edumail.vic.gov.au

Copyright in materials appearing at any sites linked to this document rests with the copyright owner/s of those materials, subject to the Copyright Act. The VCAA recommends you refer to copyright statements at linked sites before using such materials.

The VCAA logo is a registered trademark of the Victorian Curriculum and Assessment Authority.

Contents

[Explicitly teaching computational and algorithmic thinking 4](#_Toc15283715)

[Overview of the resource 5](#_Toc15283716)

[Enter the matrix 7](#_Toc15283717)

[Appendix 1 22](#_Toc15283719)

[Appendix 2 23](#_Toc15283720)

[Appendix 3 24](#_Toc15283721)

Explicitly teaching computational and
algorithmic thinking

The *Computational and algorithmic thinking – Unpacking the content descriptions* resources unpack the Victorian Curriculum F–10 Mathematics content descriptions that address computational thinking and algorithms at each level in the Patterns and algebra sub-strand of the Number and Algebra strand.

Each resource provides teachers with links between one Mathematics content description and extract from the achievement standard related to computational and algorithmic thinking and a teaching and learning activity that is designed to develop computational thinking and problem-solving skills and produce corresponding algorithms in a mathematical context. Teachers can also find excerpts from the Victorian Curriculum Mathematics and Digital Technologies glossaries in [Appendix 3](#Appendix3).

The resources have been developed with respect to teaching in the Mathematics learning area of the Victorian Curriculum and they also include suggestions how these activities could be extended to the Critical and Creating Thinking and Digital Technologies curriculums.

Teachers will find detailed ideas about how to integrate Mathematics with one or both of Digital Technologies and Critical and Creating Thinking in [Appendix 1](#Appendix1) and [Appendix 2](#Appendix2) respectively.

Overview of the resource

**Curriculum area and level:** Mathematics, Level 10

**Strand and sub-strand:** Number and Algebra, Patterns and algebra

**Content description:** Implement algorithms using data structures in a general-purpose programming language [(VCMNA334)](https://victoriancurriculum.vcaa.vic.edu.au/mathematics/curriculum/f-10#level=10&search=54186d64-6e01-4c00-a80c-a4c10108d2fc)

**Achievement standard (extract):** Students … solve problems involving linear equations and inequalities … and pairs of simultaneous linear equations and related graphs, with and without the use of digital technology.

**Title:** Enter the matrix

**Timing:** 1 or 2 lessons (approx. 50­–100 minutes)

**Description:** Students use technology and general-purpose programming language to represent and apply transformations to points in the plane. They are encouraged to experiment with the matrices and transformations that result, in order to deepen their understanding of the way matrices are used for this purpose.

 Students’ summary findings could be used to create a visual display in the classroom.

**Learning objectives:** Students can:

use a programming language to represent matrices

use matrices to transform points in the plane

describe the properties of matrices used to transform points in the plane

visually represent these transformations in the plane.

**Printable materials:** Each student or student pair will need a computer and access to the internet or software such as Mathematica.

 For this activity, the programming language used is the Wolfram language. The Victorian Department of Education and Training has a licence for the following Wolfram products:

Wolfram

SystemModeler

Wolfram Programming Lab

Mathematica

Wolfram Alpha Pro

 The licence covers these programs being used in any Victorian secondary school (not just government schools). See the [Victorian Department of Education and Training website](https://www.education.vic.gov.au/about/programs/learningdev/vicstem/Pages/wolframsoftware.aspx) to access the software and to find out how to use these programs on school computers, teacher devices and student devices.

Alternatively, any similar computer algebra system or programming language could be used. The same learning activities would still be applicable, using the syntax of the chosen programming language, such as Python, or software such as GeoGebra or Desmos.

Enter the matrix

Scaffold and support

Teachers are encouraged to use and clearly define the terms ‘matrix’ and ‘algorithm’ in this activity to reinforce the process of following a set or sequence of instructions to solve a problem (see the [Victorian Curriculum Mathematics Glossary](http://victoriancurriculum.vcaa.vic.edu.au/LearningArea/LoadFile?learningArea=mathematics&subject=mathematics&name=Mathematics%20Glossary.docx&storage=Glossary) or [Appendix 3](#Appendix3)). Teachers could choose to define these terms themselves for the class or give students the opportunity to come up with their own agreed class definitions before, during or after the activity and compare them with the definitions in the glossary.

Some key vocabulary and background information have been included below.

**Text matrix** (plural: **matrices**)

A matrix is a rectangular array of numbers. For example, the following matrix represents the fitness ratings on a scale of 0–100 for three students measured in each of four school terms (students would be represented by the row, their fitness score for each term would be recorded in the columns):

$$\left(\begin{matrix}72&78&93&81\\68&62&75&78\\66&78&72&80\end{matrix}\right)$$

The dimensions of the matrix above are 3 × 4 (read ‘three by four’) – that is, there are three rows and four columns.

Matrices are also used for representing and working with linear transformations, as we will explore.

**Representing figures using matrices**

A point in the plane with coordinates (x, y) can be represented using a matrix $\left(\genfrac{}{}{0pt}{}{x}{y}\right)$ where the upper value is the x-coordinate and the lower value is the y-coordinate.

For example, the point (1,4) in the Cartesian plane can be represented as $\left(\begin{matrix}1\\4\end{matrix}\right)$ .

The square with vertices at points (–1,4), (–1,1), (2,1), (2,4) is shown below.

ListLinePlot[{{-1,4},{-1,1},{2,1},{2,4},{-1,4}},PlotRange->{{-2,5},
{-1,5}},AspectRatio->Automatic, Filling->Automatic, AxesLabel->{x,y}]

![ListLinePlot[{{-1,4},{-1,1},{2,1},{2,4},{-1,4}},PlotRange->{{-2,5},{-1,5}},AspectRatio->Automatic, Filling->Automatic, AxesLabel->{x,y}] ]()

These points can be represented using the matrix:

$\left(\begin{matrix}-1&-1&2&2\\4&1&1&4\end{matrix}\right)$

Activity – Working with matrices in Mathematica

Student should begin the activity by defining and/or revising a matrix.

Students should then open a new notebook in Mathematica to work with matrices.

Direct the students through Parts 1–3 of this activity.

Note, in this document the indented grey Courier New text represents the input and output in Mathematic. The input text is bold and the output is not bold, as per Mathematica.

**Tips:**

* In Mathematica, to execute (run) a line of code, press SHIFT + ENTER on an input line. The output line will show below it.
* To insert a new line above or below your current line, hover above or below the line until your cursor is horizontal and then click.

Part 1: Represent a point as a matrix and plot this point on a graph

Consider the point (1,4).

When using Mathematica, the equal sign (=) will assign an expression or value to the word or letter that is on the left of the equal sign.

For example, type the input z = 2 and execute SHIFT + ENTER. The output is 2

z = 2

2

and

z + 3

5

To remove this assigned value, use the Clear command and check z no longer has the value 2.

Clear [z]

z

z

Input the point (1,4) as a matrix using curly brackets, to assign p1 = $\left(\begin{matrix}1\\4\end{matrix}\right)$

p1 = {1,4}

{1,4}

To output this in the more familiar matrix form $\left(\begin{matrix}1\\4\end{matrix}\right)$, use the MatrixForm command.

p1 / / MatrixForm

$\left(\genfrac{}{}{0pt}{}{1}{4}\right)$

Plot the point (1,4) using the ListPlot command. Set the origin to (0,0) and label the axes as x and y.

ListPlot[{{1,4}}, AxesLabel $\rightarrow $ {x,y}, AxesOrigin $\rightarrow $ {0,0}, AspectRatio $->$ 1, PlotStyle → PointSize[0.03]]

![ListPlot[{{1,4}}, AxesLabel → {x,y}, AxesOrigin → {0,0}, AspectRatio -> 1, PlotStyle → PointSize[0.03]]]()

► **Cross-curricular links**

See [Appendix 1](#Appendix1) and [Appendix 2](#Appendix2) for ways to link this activity to the explicit teaching of Digital Technologies and/or Critical and Creative Thinking.

Part 2: Use transformation matrices to transform the point p1

Matrices can be used to translate points.

To translate the point (x, y) to x + 3 and y – 2 simply add 3 to the x-coordinate and subtract 2 from the y-coordinate (or each x-coordinate and y-coordinate if translating a shape).

For example, translating (1,4) right 3 and down 2:

$\left(\genfrac{}{}{0pt}{}{1}{4}\right) \rightarrow translate \rightarrow \left(\genfrac{}{}{0pt}{}{1+3}{4-2}\right)=\left(\genfrac{}{}{0pt}{}{4}{2}\right)$

Or using Mathematica, and the point p1 = $\left(\genfrac{}{}{0pt}{}{1}{4}\right)$

p1 + {3,­-2}//MatrixForm

$\left(\genfrac{}{}{0pt}{}{4}{2}\right)$

To apply a dilation, multiply each x- and y-coordinate with the corresponding scale factor.

For example, to dilate by scale factor 3:

$$\left(\begin{matrix}1\\4\end{matrix}\right)\rightarrow scale factor of 3\rightarrow 3\left(\begin{matrix}1\\4\end{matrix}\right)=\left(\begin{matrix}3\\12\end{matrix}\right)$$

Or using Mathematica:

3p1//MatrixForm

$\left(\genfrac{}{}{0pt}{}{3}{12}\right)$

To reflect a point in a line, a reflection matrix is used.

The most common reflection matrices are:

* for a reflection in the x-axis



with the matrix $\left(\begin{matrix}1&0\\0&-1\end{matrix}\right)$

* for a reflection in the y-axis



with the matrix $\left(\begin{matrix}-1&0\\0&1\end{matrix}\right)$

* for a reflection in the line y *=* x



with the matrix $\left(\begin{matrix}0&1\\1&0\end{matrix}\right)$

To reflect a point or shape, multiply the point or shape matrix by the reflection matrix.

The following shows the effect of applying each of the above to the point (1,4).

**Tips:**

The symbol (.) is used for matrix multiplication and the MatrixForm command is used to output the answer in matrix form.

Important: Don’t assign m1 to be the matrix in matrix form. Multiplication using the (.) operator can only be done if the matrix is in curly bracket form, for example, m1 = {{1,0}, {0,1}}.

Reflection in the x-axis

m1 = {{1,0}, {0,1}}

{{1,0}, {0,–1}}

m1//MatrixForm

$\left(\begin{matrix}1&0\\0&-1\end{matrix}\right)$

**m1.p1//MatrixForm**

$\left(\genfrac{}{}{0pt}{}{1}{-4}\right)$

ListPlot[{{1,4}, {1,–4}}, AxesLabel $\rightarrow $ {x,y}, AxesOrigin $\rightarrow $ {0,0}, AspectRatio - > 1, PlotStyle $\rightarrow $ PointSize[0.03]]

![ListPlot[{{1,4}, {1,–4}}, AxesLabel → {x,y}, AxesOrigin → {0,0}, AspectRatio - > 1, PlotStyle → PointSize[0.03]]]()

Reflection in the y-axis

m2 = {{–1,0}, {0,1}}

{{–1,0}, {0,1}}

m2//MatrixForm

$\left(\begin{matrix}–1&0\\0&1\end{matrix}\right)$

**m2.p1//MatrixForm**

$\left(\genfrac{}{}{0pt}{}{–1}{4}\right)$

ListPlot[{{1,4}, {–1,4}}, AxesLabel $\rightarrow $ {x,y}, AxesOrigin $\rightarrow $ {0,0}, AspectRatio - > 1, PlotStyle $\rightarrow $ PointSize[0.03]]

![ListPlot[{{1,4}, {–1,4}}, AxesLabel → {x,y}, AxesOrigin → {0,0}, AspectRatio - > 1, PlotStyle → PointSize[0.03]]]()

Reflection in the line y = x

m4 = {{0,1}, {1,0}}

{{0,1}, {1,0}}

m4//MatrixForm

$\left(\begin{matrix}0&1\\1&0\end{matrix}\right)$

**m4.p1//MatrixForm**

$\left(\genfrac{}{}{0pt}{}{4}{1}\right)$

ListPlot[{{1,4}, {4,1}}, AxesLabel $\rightarrow $ {x,y}, AxesOrigin $\rightarrow $ {0,0}, AspectRatio - > 1, PlotStyle $\rightarrow $ PointSize[0.03]]

![ListPlot[{{1,4}, {4,1}}, AxesLabel → {x,y}, AxesOrigin → {0,0}, AspectRatio - > 1, PlotStyle → PointSize[0.03]]]()

► **Cross-curricular links**

See [Appendix 1](#Appendix1) and [Appendix 2](#Appendix2) for ways to link this activity to the explicit teaching of Digital Technologies and/or Critical and Creative Thinking.

Part 3: Experiment with combinations of transformations

Now that they are familiar with the basic transformation, let students experiment with multiplying by combinations of different transformation matrices. Ask them to predict what will happen to their point and explain why.

**Discussion prompts**

* Which of the other transformation matrices does this remind you of?
* How is it similar? How is it different?
* How has the original point been mapped to the transformed point?
* Looking at the transformation matrix, why do you think this has happened?

Question 1

What happens when the point (1,4) represented as $\left(\begin{matrix}1\\4\end{matrix}\right) $is multiplied by the matrix $\left(\begin{matrix}4&0\\0&-4\end{matrix}\right)$ ?

This is 4 times the reflection matrix $\left(\begin{matrix}1&0\\0&-1\end{matrix}\right)$ (a reflection about the x-axis) because

$$\left(\begin{matrix}4&0\\0&-4\end{matrix}\right)=4\*\left(\begin{matrix}1&0\\0&-1\end{matrix}\right)$$

Is this transformation a scaling of 4 and a reflection over the x-axis? Check using Mathematica.

m6 = {{4,0}, {0,–4}}

{{4,0}, {0,–4}}

m6//MatrixForm

$\left(\begin{matrix}4&0\\0&–4\end{matrix}\right)$

**m6.p1**

{{4}, {–16}}

ListPlot[{{4,–16}, {1,4}}, AxesLabel $\rightarrow $ {x,y}, AxesOrigin $\rightarrow $ {0,0}, AspectRatio - > 1, PlotStyle $\rightarrow $ PointSize[0.03]]

![ListPlot[{{4,–16}, {1,4}}, AxesLabel → {x,y}, AxesOrigin → {0,0}, AspectRatio - > 1, PlotStyle → PointSize[0.03]]]()

The original point (on the left) is reflected in the x-axis and is 4 times as far from the x- and y-axes.

Question 2

What happens if the point (1,4) represented as $\left(\begin{matrix}1\\4\end{matrix}\right) $is multiplied by the matrix $\left(\begin{matrix}-3&0\\0&2\end{matrix}\right)$ ?

Using Mathematica, it can be seen that this is a combination of reflection and different scaling/dilation for x and y:

m5 = {{–3,0}, {0,2}}

{{–3,0}, {0,2}}

m5//MatrixForm

$\left(\begin{matrix}–3&0\\0&2\end{matrix}\right)$

**m5.p1**//MatrixForm

$\left(\begin{matrix}-3\\8\end{matrix}\right)$

ListPlot[{{1,4}, {–3,8}}, AxesLabel $\rightarrow $ {x,y}, AxesOrigin $\rightarrow $ {0,0}, AspectRatio - > 1, PlotStyle $\rightarrow $ PointSize[0.03]]

![ListPlot[{{1,4}, {–3,8}}, AxesLabel → {x,y}, AxesOrigin → {0,0}, AspectRatio - > 1, PlotStyle → PointSize[0.03]]]()

Since the matrix for reflection in the y-axis is $\left(\begin{matrix}-1&0\\0&1\end{matrix}\right)$ , this explains why the point has been reflected in the y-axis. The x-coordinate has its sign reversed and has been scaled by 3 (so it is 3 times as far from the y-axis and to the left) – that is,
$$x\rightarrow -3x$$

The y-coordinate has been scaled by 2 (it is now twice as far up from the x-axis) – that is,

$$y\rightarrow 2y$$

Encourage students to play with these types of asymmetric transformation matrices, and encourage them to explain what has happened to their point and why they think this has happened.

**Tip:** For a revision of using matrices for transformations, see [Transformation using matrices](https://www.mathplanet.com/education/geometry/transformations/transformation-using-matrices) (Math Planet), which consists of a written summary and a video of worked examples.

► **Cross-curricular links**

See [Appendix 1](#Appendix1) and [Appendix 2](#Appendix2) for ways to link this activity to the explicit teaching of Digital Technologies and/or Critical and Creative Thinking.

Challenge and extend

Once they have worked with transformations of points, ask students to define a simple shape such as a square or triangle and apply transformations to this shape. Before they apply it, have students predict what they think their transformation matrix will do to the shape*.*

For example, consider the square represented by the matrix $\left(\begin{matrix}-1&-1&2\\4&1&1\end{matrix} \begin{matrix}2\\4\end{matrix} \right)$ .



A transformation matrix of the form $\left(\begin{matrix}-2&0\\0&2\end{matrix}\right)$ will enlarge the square (scale by 2) and reflect it in the y-axis.

Check this using Mathematica.

p2 = {{–1,–1,2,2}, {4,1,1,4}}

{{–1,–1,2,2}, {4,1,1,4}}

p2//MatrixForm

$\left(\begin{matrix}–1&-1&2&2\\4&1&1&4\end{matrix}\right)$

ListPlot[{{–1,4}, {–1,1}, {2,1}, {2,4}}, AxesLabel $\rightarrow $ {x,y}, AxesOrigin $\rightarrow $ {0,0}, AspectRatio - > 1, PlotStyle $\rightarrow $ PointSize[0.03]]

![ListPlot[{{–1,4}, {–1,1}, {2,1}, {2,4}}, AxesLabel → {x,y}, AxesOrigin → {0,0}, AspectRatio - > 1, PlotStyle → PointSize[0.03]]]()

m7 = {{–2,0}, {0,2}}

{{–2,0}, {0,2}}

m7//MatrixForm

$\left(\begin{matrix}–2&0\\0&2\end{matrix}\right)$

m7.p2//MatrixForm

$\left(\begin{matrix}2&2&-4&-4\\8&2&2&8\end{matrix}\right)$

ListPlot[{{2,8}, {2,2}, {-4,2}, {-4,8}}, AxesLabel $\rightarrow $ {x,y}, AxesOrigin $\rightarrow $ {0,0}, AspectRatio - > 1, PlotStyle $\rightarrow $ PointSize[0.03]]

![ListPlot[{{2,8}, {2,2}, {-4,2}, {-4,8}}, AxesLabel → {x,y}, AxesOrigin → {0,0}, AspectRatio - > 1, PlotStyle → PointSize[0.03]]]()

The square is twice as high and twice as wide, and has been reflected in the y-axis, as expected.

Encourage students to research to find other uses for matrices, for example, in computer science. Why might a matrix be a good way of storing data for these particular uses?

Reflection

Students should experiment with their transformation matrices and then share their results with others to compare their solutions.

With the class, the teacher could ask the students to discuss the challenges they met and discuss different strategies that students used to overcome these.

* What were some challenges you faced in this activity?
* How did you overcome these (what were some strategies)?
* What were some skills you have learnt that helped in a similar situation?
* Can you think of other problems where a matrix might be used to help solve a complicated problem?

With the class, in small groups or individually, the teacher should ask students to reflect on their algorithmic thinking. Ask students:

* How does the transformation matrix work in each activity?
* What do transformation matrices do well? Why?
* What don’t they do well? Why?
* What do you think was the purpose of this investigation?

**Tip:** Reflection is a very important part of any computational thinking–focused activity because it encourages students to consider the different aspects of the task, such as defining the problem, selection of tools and processes, problem-solving, teamwork and verifying their solution. This helps students reflect on the process of their own learning (meta-learning) and how the skills and tools they have used might be transferred to other contexts.

Additional teaching resources

The [Code.org website](https://code.org/educate/curriculum/high-school) is a great way to introduce students to a range of programming languages, with different activities for students with different levels of assumed knowledge. Courses can be taken as individual lessons or as a sequence. Encourage students to explore and select an activity or course that interests them and to work on their algorithmic thinking by programming in a general-purpose programming language.

Appendix 1

Suggestions for explicitly teaching Digital Technologies (stimulus only)

**Curriculum area:** Digital Technologies

**Strand:**

**Band:** Levels 9 and 10

**Content description:** Design algorithms represented diagrammatically and in structured English and validate algorithms and programs through tracing and test cases [(VCDTCD052)](http://victoriancurriculum.vcaa.vic.edu.au/Curriculum/ContentDescription/VCDTCD052)

**Achievement standard (extract):** They design … algorithms … and test modular programs.

**Suggestions that link to the activity:**

* Creating algorithms that use various functions and data structures.
* Testing the expected output of algorithms using trace tables and desk checking if necessary in order to make modifications and record results.

See also [Unpacking Digital Technologies Content Descriptions](https://www.vcaa.vic.edu.au/Documents/viccurric/digitech/Unpacking_the_Content_Descriptions/Unpacking_Digital_Technologies_Content_Descriptions-9-10.docx) for Levels 9 and 10.

Appendix 2

Suggestions for explicitly teaching Critical and Creative Thinking (stimulus only)

**Curriculum area:** Critical and Creative Thinking

**Strand:** Questions and possibilities

**Band:** Levels 9 and 10

**Content description:** Investigate the characteristics of effective questions in different contexts to examine information and test possibilities ([VCCCTQ043](http://victoriancurriculum.vcaa.vic.edu.au/Curriculum/ContentDescription/VCCCTQ043))

**Achievement standard (extract):** … students construct and evaluate questions, including their own, for their effectiveness.

**Suggestion that links to the activity:**

* Constructing questions that facilitate discussion of the properties of the matrices chosen that affect the type of transformation of the points in the plane.

**Curriculum area:** Critical and Creative Thinking

**Strand:** Reasoning

**Band:** Levels 9 and 10

**Content description:** Investigate the nature and use of counter examples structured as arguments ([VCCCTR048](http://victoriancurriculum.vcaa.vic.edu.au/Curriculum/ContentDescription/VCCCTR048))

**Achievement standard (extract):** Students structure complex valid arguments.

**Suggestion that links to the activity:**

* Using reasoning to construct a counterexample – for example, ‘If I combine these transformation matrices, then the transformation is/is not the same (equivalent) if found by combining other transformation matrices’ – to make students aware of what the point of finding a counterexample is.

**Curriculum area:** Critical and Creative Thinking

**Strand:** Meta-Cognition

**Band:** Levels 9 and 10

**Content description:** Critically examine their own and others thinking processes and discuss factors that influence thinking, including cognitive biases ([VCCCTM051](http://victoriancurriculum.vcaa.vic.edu.au/Curriculum/ContentDescription/VCCCTM051))

**Achievement standard (extract):** Students identify, articulate, analyse and reflect on their own and others’ thinking processes.

**Suggestion that links to the activity:**

* Researching, comparing and reflecting on thinking processes used by others to create the required transformations

Appendix 3

Excerpts from the Victorian Curriculum Mathematics and Digital Technologies glossaries

There are some commonalities in the terms used when explicitly teaching computational and algorithmic thinking in Mathematics and Digital Technologies; however, there are also some subtle but important differences in the definitions of terms. Some of these common terms and their definitions are listed below, under the two different curriculum areas.

If you are going to explicitly teach Mathematics, please refer to the [Victorian Curriculum Mathematics Glossary](http://victoriancurriculum.vcaa.vic.edu.au/LearningArea/LoadFile?learningArea=mathematics&subject=mathematics&name=Mathematics%20Glossary.docx&storage=Glossary). If you are also going to explicitly teach Digital Technologies, refer to the [Victorian Curriculum Digital Technologies Glossary](https://victoriancurriculum.vcaa.vic.edu.au/LearningArea/LoadFile?learningArea=technologies&subject=digital-technologies&name=Digital%20Technologies%20Glossary.docx&storage=Glossary).

Mathematics

Algorithm

An **algorithm** is a process that can be carried out mechanically, using a well-defined set of instructions, to perform a particular task or solve a type of problem. Examples of mathematical algorithms include processes for tasks such as ordering a set of numbers from smallest to largest, multiplying many-digit decimal numbers, factorising linear expressions, determining which of two fractions is larger, bisecting an angle, or calculating the mean of a set of numbers.

Algorithmic thinking

**Algorithmic thinking** is the type of thinking required to design, test and evaluate problem-solving processes in a systematic way, using algorithms.

Array

An **array** is an ordered collection of objects or numbers. For example, the following is a triangular array of dots:



A rectangular array of numbers is also called a **matrix** (plural: matrices). For example, the following matrix represents the fitness ratings on a scale of 0 – 100 for three students measured in each of four school terms (students would be represented by the row, their fitness score for each term would be recorded in the columns):



Coding

A process by which algorithms are represented for implementation. For computers, this is done using a coding language such as block coding, C++, JavaScript, Python, Wolfram Language.

Computational thinking

In this context, computational thinking is considered to be linked to algorithmic thinking. This type of thinking is usually considered specific to computers which involves solving problems, designing systems and implementation.

Enlargement (dilation)

An **enlargement** is a transformation that scales up (or down) a figure in which the corresponding lengths in the transformed figure are in proportion to the original figure. The relative positions of points are unchanged and the two figures are similar.

In the diagram below triangle A′B′C′ is the image of triangle ABC under the enlargement with enlargement factor 2 and centre of enlargement O.



Sequence (number)

A **sequence** is an ordered set of elements such as numbers, instructions or objects. From an algorithmic point of view, a sequence is an ordered set of instructions or actions.

Unplugged

A commonly used term for computational thinking activities carried out without digital technology. “Unplugged” representations of algorithms may include structured mathematical processes, English representations (steps) or flowcharts.

Digital Technologies

Algorithm

A description of the steps and decisions required to solve a problem. For example, to find the largest number in a list of positive numbers:

1. Note the first number as the largest
2. Look through the remaining numbers, in turn, and if a number is larger than the number found in 1, note it as the largest.
3. Repeat this process until complete. The last noted number is the largest in the list.

Flowcharts are often useful in visualising an algorithm.

Computational thinking

A problem-solving method that involves various techniques and strategies in order to solve problems that can be implemented by digital systems, such as organising data logically, breaking down problems into components, and the design and use of algorithms, patterns and models.

Structured English

The use of the English language to describe the steps of an algorithmin clear, unambiguous statements that can be read from start to finish. The use of keywords, such as START, END, IF, UNTIL, provides a syntax similar to that of a programming language to assist with identifying logical steps necessary to properly describe the algorithm.

An example of the use of structured language can be demonstrated using the following problem:

Description of the problem:

Describing the decision a person makes about how to get to a destination based on the weather and the distance from their current location to their destination.

Structured English example:

START

IF it is raining outside THEN

 Catch the bus

ELSE

 IF it is less than 2km to the destination THEN

 Walk

 ELSE IF it is less than 10km to the destination THEN

 Ride a bicycle

 ELSE

 Catch the bus

 ENDIF

ENDIF

END

The Structured English description can easily be translated into code using a programming language and accurately captures the logical elements that must be followed to answer the question posed.