

Week 4					
Lesson	Spec ref	Lesson summary	Lesson content	Lesson resources	Transferable skills
1	3.3.2 3.3.3 3.3.4	Data storage and compression: RLE	<p>Show the YouTube video ‘Run Length Encoding Visualization’ to introduce the concept of RLE. http://www.youtube.com/watch?v=ypdNscvym_E</p> <p>Give students the opportunity to experience RLE compression in action by decompressing a 4-bit colour image encoded using RLE. (Activity 4.1.1)</p> <p>Explain how to calculate the size (in bytes) of the uncompressed and the compressed files, pointing out the small size reduction. (Activity 4.1.2)</p> <p>Point out that the advantage of a smaller file is gained at the expense of the computer having to carry out more processing – the file has first to be compressed and then it has to be decompressed before it can be used. This is especially true in this case, where compression only saves one byte. Ask students to consider whether the file would compress more or less if colours were represented by 4-bit codes. Introduce students to the RLE calculator: http://mathcelebrity.com/runlencode.php. Give them an opportunity to try it out and get to grips with how it works. (Activity 4.1.3)</p> <p>Homework: Ask students to complete Activity 4.1.4.</p>	<p>Week 4, Lesson 1 activities</p> <p>YouTube video: ‘Run Length Encoding Visualisation’</p> <p>RLE calculator</p>	<p>Critical thinking</p> <p>Self-direction</p>
2	1.1.6 3.3.3	Data storage and compression: RLE	<p>Ask students to work in pairs to write and test an RLE program using the algorithm they completed for the last homework. (Activity 4.2.1)</p> <p>Homework: Ask students to complete the compression summary sheet. (Activity 4.2.2).</p>	<p>Week 4, Lesson 2 activities</p> <p>IDLE editor for Python</p>	<p>Problem solving</p> <p>Collaboration</p>

Week 4, Lesson 1 activities

Activity 4.1.1

This series of binary numbers represents a compressed image.

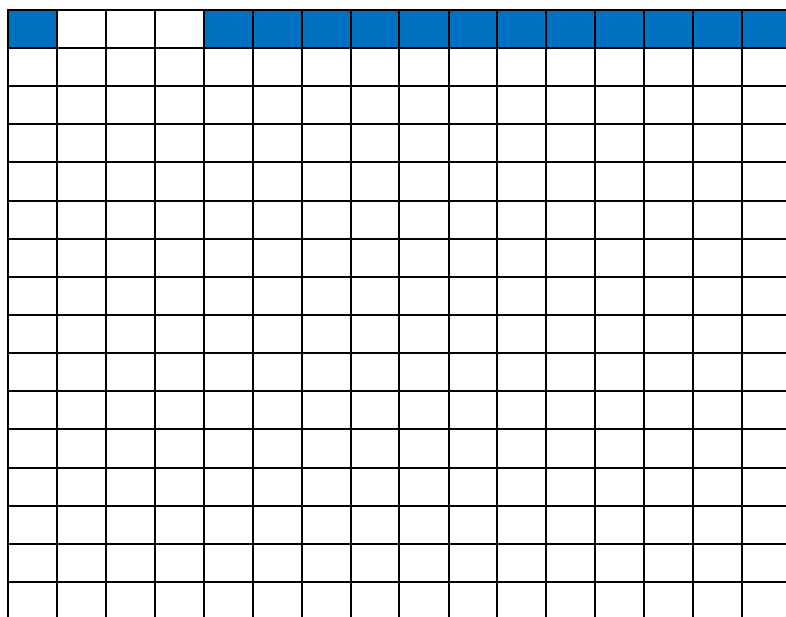
Line 1	00 0001 01 0011 00 1100
Line 2	00 0001 01 0011 00 1100
Line 3	00 0001 01 0011 00 0100 10 0010 00 0100 10 0001 00 0001
Line 4	00 0111 10 0010 00 0100 10 0010 00 0001
Line 5	01 0010 00 0101 11 0011 00 0011 10 0010 00 0001
Line 6	01 0010 00 0011 11 0110 00 0001 10 0011 00 0001
Line 7	00 0100 11 0010 01 0001 11 0101 10 0011 00 0001
Line 8	00 0001 01 0001 00 0001 11 1001 10 0011 00 0001
Line 9	00 0011 11 1001 10 0011 00 0001
Line 10	00 0001 01 0001 00 0010 11 1000 10 0011 00 0001
Line 11	00 0101 11 0110 00 0001 10 0011 00 0001
Line 12	00 0111 11 0011 00 0011 10 0010 00 0001
Line 13	00 0001 11 0001 00 0010 11 0001 00 0010 10 0010 00 0100 10 0010 00 0001
Line 14	00 0001 11 0001 00 0001 11 0001 00 0100 10 0010 00 0100 10 0001 00 0001
Line 15	00 0010 11 0001 00 1101
Line 16	10 10000

The first pair of binary numbers represents a colour code as follows:

00 = blue 01 = white 10 = yellow 11 = green

The second set of binary numbers represents the run length of the colour. So, for example, 00, 1101 represents 13 blue pixels.

Assuming each of the squares in the grid below represents one pixel, shade in the squares. The first line of pixels has been done for you.



Activity 4.1.2

Use this image to complete these calculations.

How many bytes are required to represent the uncompressed file? Hint: number of squares in grid x 2 / 8	
How many bytes are required to represent the RLE encoded file? Hint: number of codes x number of bits in each code / 8	
How much storage space have you saved?	

Extension

Imagine that an image contains:

- 1,024 pixels (a grid of 32 pixels by 32 pixels)
- 255 different colours (represented by 8 bits)

The image has been encoded using RLE, producing 116 binary codes comprising 8 bits (to represent the 255 colours) and 5 bits (to represent the colour run length).

Using this information, calculate the following.

How many bytes are required to represent the uncompressed file?	
How many bytes are required to represent the RLE encoded file?	
How much storage space have you saved?	

Activity 4.1.3

Encode these text strings using RLE.

Text string	Answer
AAAABBBBBBBBBBCADDDDEEFFFFFF FF	
ABCABCABCABCABCABCABCABCAB CS	
BBGGYYAACCFEEBBGGYYAACCF EE	

Which one compresses the most?	
Why is this?	
Describe in English the process the RLE calculator follows to encode a piece of text.	

Activity 4.1.4 (homework)

1. Explain how the RLE compression algorithm works.
2. Here is some data used to represent an image. Each pixel is encoded as a character.
AADACCEFABAAECFGBDGE
Explain why a RLE algorithm may not be appropriate for encoding this image.
3. Here is a partly completed RLE algorithm expressed as a written description. Fill in the gaps to complete it.
 1. Start with the first character in the string.
 2. Write down the number 1.
 3. Compare the first character with the next character on the right.
 4. If they are the same, _____.
 5. If they are not the same, _____.
 6. Move on to the next character on the right.
 7. Go back to step 2 and repeat until you reach the end of the string.
 8. _____.

Week 4, Lesson 2 activities

Activity 4.2.1

Implement the RLE compression algorithm in Python.

Test out your program using these data strings:

- AAAABBBBBBBBBBCADDDDEEEEEFFFFF
- ABCABCABCABCABCABCABCABCABC
- BBGGYYAACCFEEBBGGYYAACCFEE

Activity 4.2.2 (homework)

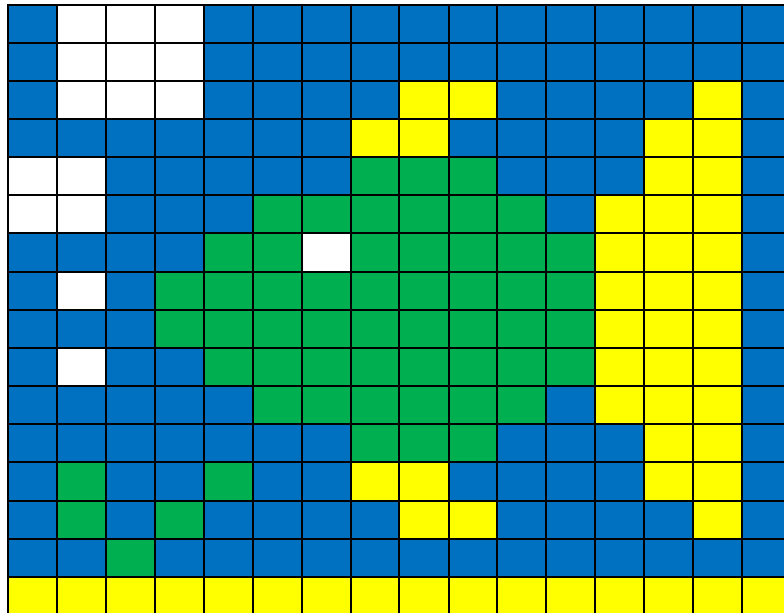
Lossless and lossy compression summary sheet

Explain what is meant by lossless compression.	
What type of data can be compressed using a lossless compression algorithm?	
Describe how a lossless RLE algorithm works.	
Some lossless compression algorithms use a lookup table. Explain what the lookup table is for.	
Explain how lossy compression differs from lossless compression.	
Explain why lossy compression is usually used for media files.	
Outline the process of compressing an audio file using a lossy compression algorithm.	
Outline the process of compressing a bitmap image using a lossy compression algorithm.	

Week 4, Lesson 1 solutions

Activity 4.1.1

Solution should look like this:



Activity 4.1.2

How many bytes are required to represent the uncompressed file? Hint: number of squares in grid x 2 / 8	$256 \times 2 / 8 = 64 \text{ bytes}$
How many bytes are required to represent the RLE encoded file? Hint: number of codes x number of bits in each code / 8	$84 \times 6 / 8 = 63 \text{ bytes}$
How much storage space have you saved?	1 byte

Extension:

How many bytes are required to represent the uncompressed file?	$= \text{number of squares in grid} \times 8 / 8$ $= 1024 \times 8 / 8$ $= 1024 \text{ bytes}$
How many bytes are required to represent the RLE encoded file?	$= \text{number of codes} \times \text{number of bits in each code} / 8$ $= 116 \times 13 / 8$ $= 189 \text{ bytes}$
How much storage space have you saved?	835 bytes

Activity 4.1.3

Text string	Answer
AAAABBBBBBBBBBCADDDDEEFFFFFFF FF	4A9BCA4D2E8F
ABCABCABCABCABCABCABCABCAB CS	ABCABCABCABCABCABCABCABCAB CS
BBGGYYAACCCFFEEBBGGYYAACCCF EE	2B2G2Y2A2C2F2E2B2G2Y2A2C2F2E

Which one compresses the most?	The first one.
Why is this?	Because it contains the longest run lengths.
Describe in English the process the RLE calculator follows to encode a piece of text.	The calculator looks at the first character and counts this as 1. It then compares the next character to the right with the first character. If they are the same, it adds 1 to the number. If they are different, it puts the letter. It then starts counting again with the next letter on the right.

Activity 4.1.4 (homework)

1. Explain how the RLE compression algorithm works.
The algorithm works by identifying repeating data and replacing it with a number representing the run length of that item.
2. Here is some data used to represent an image. Each pixel is encoded as a character.

AADACCEFABAAECFGBDGE

Explain why a RLE algorithm may not be appropriate for encoding this image.

The sequence has very few repeated patterns, therefore using a RLE will not lead to a significant reduction in the file size.

3. The completed algorithm:
 1. Start with the first character in the string.
 2. Write down the number 1.
 3. Compare the first character with the next character on the right.
 4. If they are the same, add 1 to the number you have written down.
 5. If they are not the same, write down the character.
 6. Move on to the next character on the right.
 7. Go back to step 2 and repeat until you reach the end of the string.
 8. Write down the last character in the string.

Week 4, Lesson 2 solutions

Activity 4.2.1

Program solution provided in ProgCode folder.

```
#Activity 4.2.1

def run_length_encoding(data):
    rleCode = ""
    length = len(data)
    if length == 0:
        rleCode += ""
    elif length == 1:
        rleCode += str(length) + data
    else:
        count = 1
        index = 1
        while index < length:
            # Checks if it is the same letter.
            if data[index] == data[index - 1]:
                count += 1
            else:
                rleCode += str(count) + data[index - 1]
                count = 1
            index += 1
        # rleCode += data[index - 1] + str(count)
        rleCode += str(count) + data[index - 1]
    return rleCode

#Main program
yourData = input("Enter the data you want to compress:")
compressed = run_length_encoding(yourData)
print("\n", compressed)
```

Activity 4.2.2 (homework)

Explain what is meant by lossless compression.	Lossless data compression reduces the size of files in such a way that the original data can be perfectly reconstructed from the compressed data – nothing is lost.
What type of data can be compressed using a lossless compression algorithm?	Lossless data compression works best on files containing strings of repeating data, whether the repeating data is characters, strings of letters or words.
Describe how a lossless RLE algorithm works.	A lossless RLE algorithm is a simple compression algorithm in which runs of data (sequences in which the same data value occurs in many consecutive data elements) are stored as a single data value and count, rather than as

	the original run. So, for example, FFFFFFFFFF would be represented as 12F.
Some lossless compression algorithms use a lookup table. Explain what the lookup table is for.	In a lookup table, a number is assigned to repeated words. Then when the compression algorithm is run, the words listed in the lookup table are replaced in the file by a number. A considerable size reduction can be achieved by using a lookup table and the original file can still be reconstructed without any loss of quality.
Explain how lossy compression differs from lossless compression.	Lossy data compression permanently discards some of the original data. It exploits the fact that human beings cannot detect subtle differences in sounds and colours. Key data is retained and less important data is discarded when lossy data compression takes place.
Explain why lossy compression is usually used for media files.	The data in photographs and audio files contains differences too subtle for the human eye or ear to detect. Therefore, lossy compression can drastically decrease the file size by discarding some of the data, while the quality is still acceptable to human beings.
Outline the process of compressing an audio file using a lossy compression algorithm.	The lossy compression algorithm analyses the data within the audio signal. It retains the key data and removes the non-audible or less audible components of the signal.
Outline the process of compressing a bitmap image using a lossy compression algorithm.	The lossy compression algorithm divides the bitmap image into blocks of 8x8 pixels. It then analyses the data within the 8x8 pixel block and ranks it according to its importance to visual perception. The lossy compression algorithm retains the key data and discards the less important data. It achieves this by replacing the colour values of some of the pixels.