Sonic Pi

Lesson 4

Data Structures & Algorithms

Designed by
Sam Aaron & Carrie Anne Philbin

Data Structures & Algorithms

Although the world is constructed from many tiny pieces, this isn't how we see it. Instead, we perceive useful structures and aggregations of these pieces. Rather than individual atoms or molecules, we see trees, people, bicycles, etc. In our previous lessons the information we have dealt with has been simple numbers - numbers for notes, numbers for time, even numbers for chance. Data structures allow us to compose and aggregate these simple data into useful structural forms. One of the most simple of these aggregate structures is the list which we will discuss in this lesson. Once we have structures like the list, we can start using a useful library of mini programs which can perform operations such as sorting and shuffling the list.

Learning Objectives	Numbers are simple elements of data.
	We can model both time and notes with numbers.
	Numbers can be aggregated into data structures such as lists.
	Algorithms are a series of steps or instructions for solving a problem such as sorting and shuffling a list of numbers.
	Many useful algorithms have already been implemented and are made available to use through programming libraries.

Teaching Progression

For the majority of the lesson, it is suggested that work is carried out in pairs. Each pair should have access to the standard equipment described below. In addition, is suggested that you have your own setup connected with a speaker for the demonstration sections.

Sonic Pi Equipment

- A Raspberry Pi with the Sonic Pi software installed per pair;
- A keyboard and mouse connected to the RPi per pair;
- A monitor connected to the RPi per pair;
- A headphone splitter connected to the RPi audio jack per pair;
- A pair of headphones connected to the splitter per pupil.

Equipment

- Number cards.

Lesson Summary

- Numbers as a kind of data.
- Introduction to lists.
- Playing a list of numbers.
- Introduction to sorting
- Sorting and shuffling a list of numbers

Starter

Pupils are first invited to set up and connect their Raspberry Pi hardware.

Main/development

1. Pupils should be asked what kind of information they have been using in the previous lessons i.e. the things they've been able to change. This is the time to sleep and the note to play. Ask them what *kind* of information this is - they're all numbers. Explain that a number is a kind of data, and there are many others that computers may understand.
2. Explain to the pupils that although numbers are one of the most useful forms of data, it's also very handy to represent a list of numbers. Ask them if they can think of any things that could be represented by a list of numbers (i.e. the finish times of runners in a race, the numbers drawn in the lottery, the notes of a bass line.)
3. Invite the pupils to enter the following code into a new worksheet and press play:

```
play_pattern [40,25,45,25,25,50,50]
```

4. Explain that `play_pattern` is similar to `play` except instead of taking a *number* representing the note to play, it takes a *list* of numbers to play one after each other. This list of numbers has special *syntax* to tell the computer that it's a list. Firstly it has to start with a `[` and end with a `]`. Secondly, each number in the list has to be separated from the other numbers with a comma.
5. Invite the pupils to write their own lists - choosing different numbers and different lengths of numbers i.e.

```
[43, 24, 60, 57, 30]
[60]
[48, 48, 48, 60]
```

6. Once they've had a short play with this, invite them to form a line and hand out the number cards (in no particular order) so each pupil in the line holds just one card. Explain that they have formed a list of numbers and that there are useful things that you can do with a list of numbers. One useful thing is sorting the numbers numerically, so that the smallest numbers are first, and the largest last. Introduce the word algorithm as a method for solving such problems.
7. Next, explain that we will explore a simple sorting algorithm: bubble sort. Start at the left hand side of the line and ask the first two pupils to compare their numbers. If they are in the right order do nothing, otherwise ask the pupils to swap. Then continue to the second and third pupils and compare and swap again if necessary. Continue down the line. If at least one pair has swapped, start at the beginning of the line and repeat. If no pairs have swapped, the list is sorted.

8. Explain that most programming languages provide many such algorithms to make programming easier and to reduce the amount of work you have to do as a programmer. Ask the pupils to type the following code:

```
play_pattern [40,25,45,25,25,50,50].sort
play_pattern [40,25,45,25,25,50,50].shuffle
```

9. Pupils are invited to play around with the constructs of this lesson in addition to everything they've learned so far to design simple a musical program.

Plenary

Ask the pupils to consider how they might design an algorithm to shuffle a list. Another thing to consider is the existence of algorithms other than sort and shuffle.

All students are able to...	Know that numbers represent data/ information. Use a list. Use sort and shuffle to manipulate a list.
Most students are able to...	Understand that lists are aggregate data structures. Understand that algorithms are solutions to problems such as sorting a list.
Some students are able to...	Describe the steps for an implementation of sort (i.e. bubble sort). Describe other data algorithms than sort and shuffle which are useful for lists (i.e. repeat, split in half, reverse).