**Note the music**

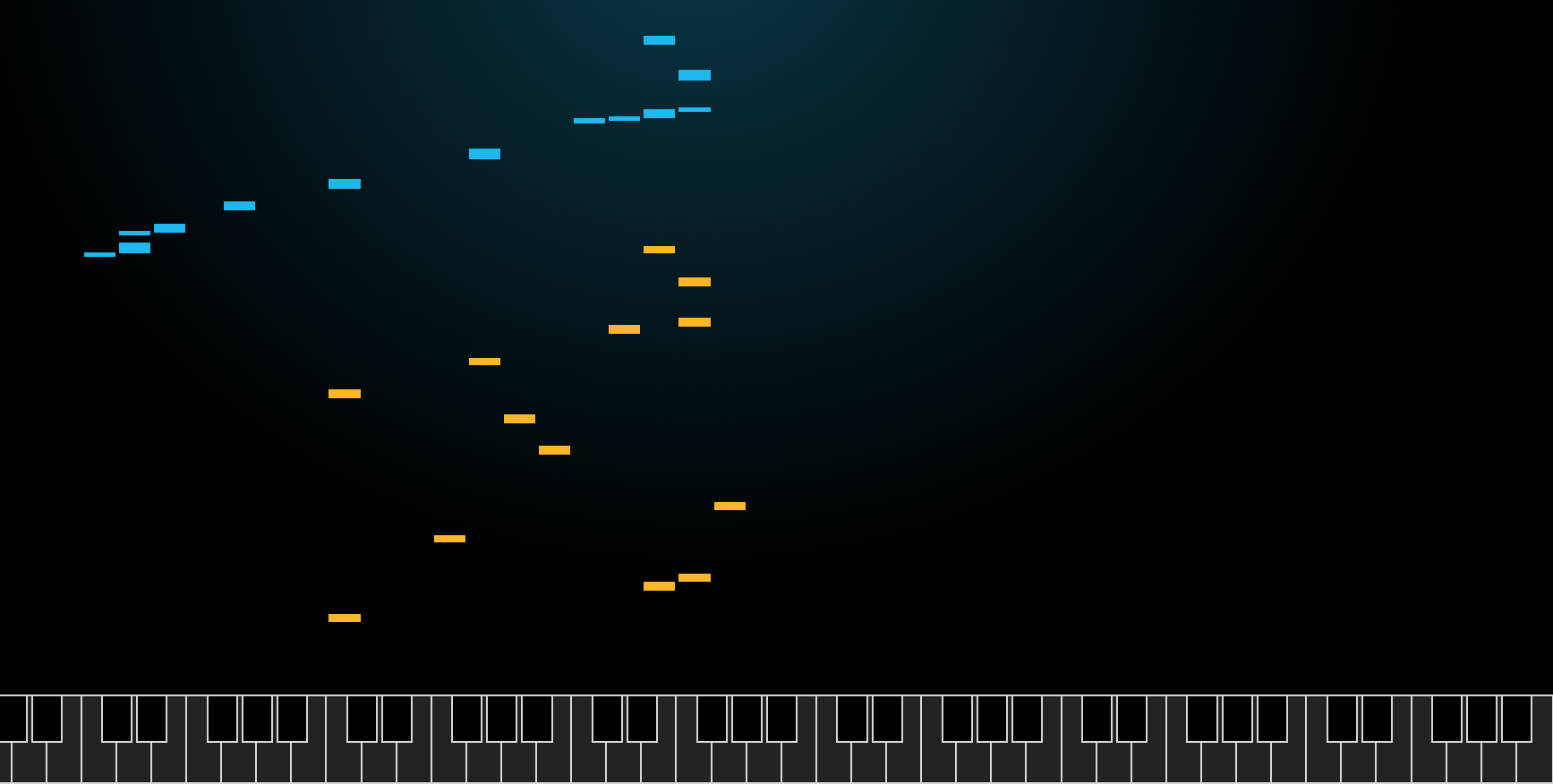
We can program a computer to play music. Conventionally this is done by hard coding, which is the process of coding all possible expected behaviours. Alternatively, we can train an artificial intelligence (AI) computer about what notes go well with others, so it can play a duet with a human musician. Students can make their own instrument that plays a particular note for a set beat, for example when a piano key is pressed or a guitar string is strummed or a drum skin is hit. Alternatively, students can incorporate the random function to mimic AI.

This set of lesson ideas was developed in collaboration with the [Digital Technologies Institute](https://www.digital-technologies.institute/) in Melbourne, Victoria.

**Year levels**: 3–4, 5–6

# Preliminary notes

The image below shows a view of the AI based tool (application) used in the ‘Plugged’ section below: [AI Duet](https://experiments.withgoogle.com/ai/ai-duet/view/). View the short video that explains this system (accessed by clicking on “how it works”) – it explains the AU used by the system in a simple and fun way.



Select the Play button to launch the AI app. At the bottom of the screen a row of virtual piano keys enables the user to play notes of the piano by using the mouse to select a key. Alternatively the user can select a letter on the keyboard to play notes. If using a tablet device the user can touch the virtual piano keys directly on screen.

The keys move from lower notes on the left to higher notes on the right.

When you play a note (touch a key), blue rectangles appear above the keyboard that represent the keys (notes) played by the user of the application.

The yellow rectangles (notes) are those played in response by the artificial intelligence.

# Suggested steps

### Unplugged activity

Music is linked to the Digital Technologies curriculum in many ways. You can teach the DT curriculum using music both unplugged and using computers such as this Google AI Duet. For example, music notation is all about codes—codes which tell the musician (or software program) what to do. Sound is a form of data that can be represented in different ways. Also, a song is a program, a list of instructions—for example, play a song and ask your students to identify iteration (loops) in it.

1. Explore beat and representation of musical notation using Lego building bricks for example as described in this [blog article](https://www.letsplaykidsmusic.com/lego-beats-music-manipulatives/). If you search on ‘Lego + musical notation’, you will find are several other articles. You may also want to show videos explaining beats (such as <https://www.youtube.com/watch?v=y46FuaUMtBw>). Here are some related questions for your students.
   * Why is beat important in music?
   * Can you represent quarter notes, half notes and a rest?
   * How are musical notes used to represent sound?
2. Provide students with a range of simple sequences of notes from familiar tunes, songs or rhythms. These may include celebration songs, [nursery rhymes](https://www.letsplaykidsmusic.com/twinkle-twinkle-little-star-easy-piano-music/), culturally relevant examples and examples from pop culture. If possible, have students use a musical instrument with notes highlighted. Some questions for your students:
   1. How is the music represented? How does that help you play the music?
   2. How did you use listening to help you play the tunes and make sure it sounded correct?
   3. Do you think a musical score is like computer code? Why / why not?
   4. Can you order the notes in a scale C, D, E, F, G, A, B, C?

**Students with a vision impairment** would benefit from adjustments to make the tasks accessible, for example students with:

* low vision would benefit from larger font for musical representations
* low contrast vision may benefit from acetate or color filters placed over a printed page of the musical representations
* no vison may benefit from braille-embossed print or tactile graphics (notes represented with different textures).

Depending on the vision impairment, students may need to work with a sighted partner to play the musical instrument.

**Students with a hearing impairment** would benefit from adjustments to make the tasks accessible. Here are some examples.

* Teachers use hearing assistive technology systems (HATS) such as an FM System or similar sound-field system, so that their voice is distributed around the classroom when they talk into a microphone.
* Teachers use/select closed captioning (CC) on any programming instructional videos, provide transcripts and repeat instructions, and have the student sit in close proximity to them.

### Plugged activities

In this activity we develop students’ programming skills using a musical context. Students will explore ways to program musical notes to create their own musical instrument to play, or play a duet between two instruments. To mimic AI, the program will apply a random change to the key and duration the musical notes are played. Finally, students compare their program to an AI tool that responds to a human, playing notes on a keyboard.

These activities use Scratch 3.0 and the additional ‘Music’ blocks.

This sequence of plugged activities has been ordered by least to most difficult and labeled in terms of ‘Levels’ (e.g. “Level One”, “Level Two”, etc.) as you may wish to use this terminology to students as they progress through each activity. Depending on the abilities of students, you may wish to start at a higher level.

***The activity has been levelled to enable differentiation.***

**Level One: Explore an AI powered computer tool**

Explore an AI powered computer tool [AI Duet](https://experiments.withgoogle.com/ai/ai-duet/view/) to see how a computer responds to a sequence of notes you play on a keyboard.

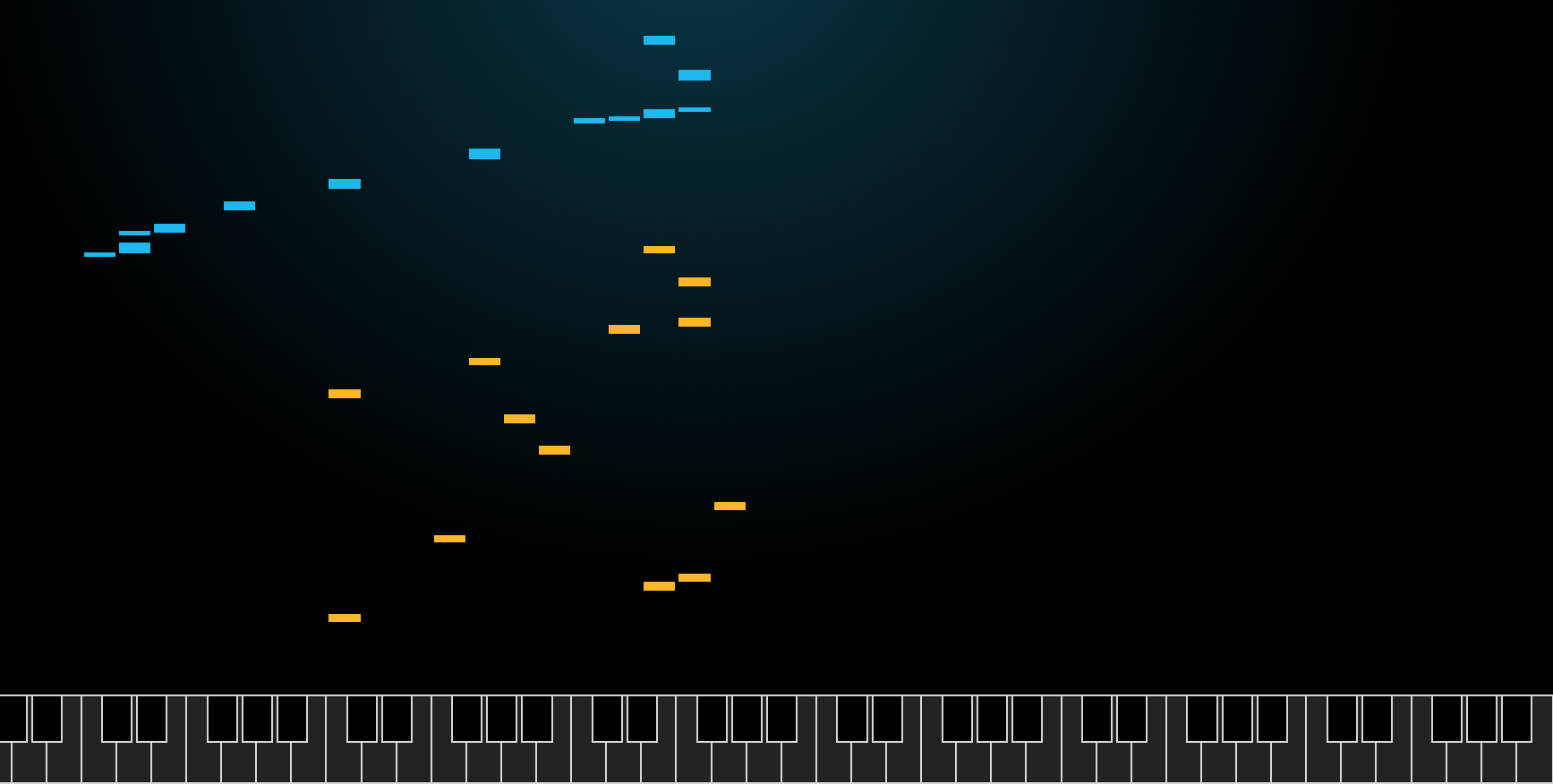


Image: AI Duet

Discuss the ways in which the AI responded to your playing in AI Duet.

1. Did you notice any patterns in the way it played?
2. Based on what you played, how do think the AI might react?
3. Could you learn from the AI?

**Level Two: Practice playing a sequence of notes**

Students can learn basic piano songs such as Chopsticks by watching a video (such as this one here: <https://www.youtube.com/watch?v=IVtw4DWnQ5w>) and following along by:

* + - 1. Using a musical instrument with notes highlighted.
      2. Using a keyboard/piano app such as Garageband on a tablet.
      3. Using a computer and website with a virtual piano (such as this one: <https://pianu.com/chopsticks-easy-piano>)
      4. Using a A4 or A3 printed piano handout (such as one from here: <http://www.piano-keyboard-guide.com/piano-keyboard-template.html>)

**Students with limited fine motor skills** would benefit from additional time to practice playing notes to compose a melody. It is recommended that they practice on musical instruments or touch-based devices so that they can get instant feedback (verses using a paper piano handout or a silent musical device).

**Level Three: Practice programming notes using Pencil Code**

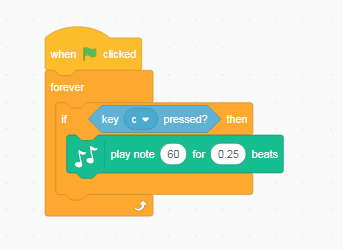
Pencil code is an interactive website to create simple text-based code. [Pencil code: Let’s jam](https://gym.pencilcode.net/jam/#/jam/firstnote.html) is specifically designed to help to make music using code and includes sample melodies and songs which can be heard and modified.

* + - 1. You may want to give students some time to explore the site and its different examples (remind students they need to click the Play symbol to run the code).
      2. Have students follow the provided challenges of ‘First Notes’, ‘A Simple Tune’, ‘Count Beats’, ‘Make Chords’ (top left corner) to change the example codes.
      3. Students can also use Pencil Code to play a virtual piano using the computer keyboard by selecting ‘Virtual Keyboard’. This will be good practice for the next level, which uses

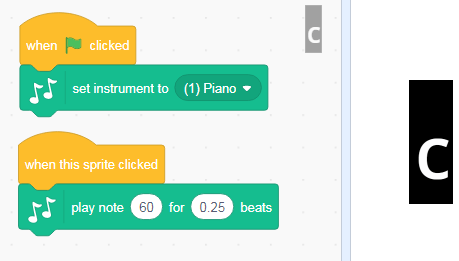
**Level Four: Program a keyboard interface (skill level: easy)**

Ask students how they could use a visual programming language such as Scratch to create music. Discuss and list the ideas. (It may help to display the Scratch programming environment for the class as you do this.) Some ideas may include:

* 1. Use a keyboard command of if/then blocks to play a sound. For example, if ‘C’ on a keyboard is pressed, play a ‘C’ note for a set beat. Note the “forever’ block is used to check this throughout the program.



* 1. Create a sprite as a piano key. Use the **when this sprite clicked** block to play a note for a set beat. Each key is a different sprite. In this example the key is set to ‘C’.



The code for this project has been provided for students to explore and remix.

**Ideas for remix**

Students could:

* change the notes by an octave either higher or lower
* double the number of keys on screen and program each key as a sprite to the correct note
* create their own interface for a different instrument and use a similar approach to programming their instrument
* use the basic idea to create a new use; for example, a quiz to match the sound with the correct note.

Sample code: Scratch [Basic piano](https://scratch.mit.edu/projects/336280504)

**Level Five: Program a duet between two instruments (skill level: medium)**

This option requires some more programming skills. Create two sprites that play a sequence of notes, responding to each other’s playing. In this case, **broadcast message** is used to trigger a response for the sprite to play. Each sprite can be a different instrument.

Step 1.

Create just one sprite that plays a sequence of notes as shown in sprite playing Melody 1. Note the broadcast block will be used to start the Sprite created in Step 2. Remember to add a ‘Hide’ block otherwise sprite 1 will continue to show while Sprite 2 is playing.

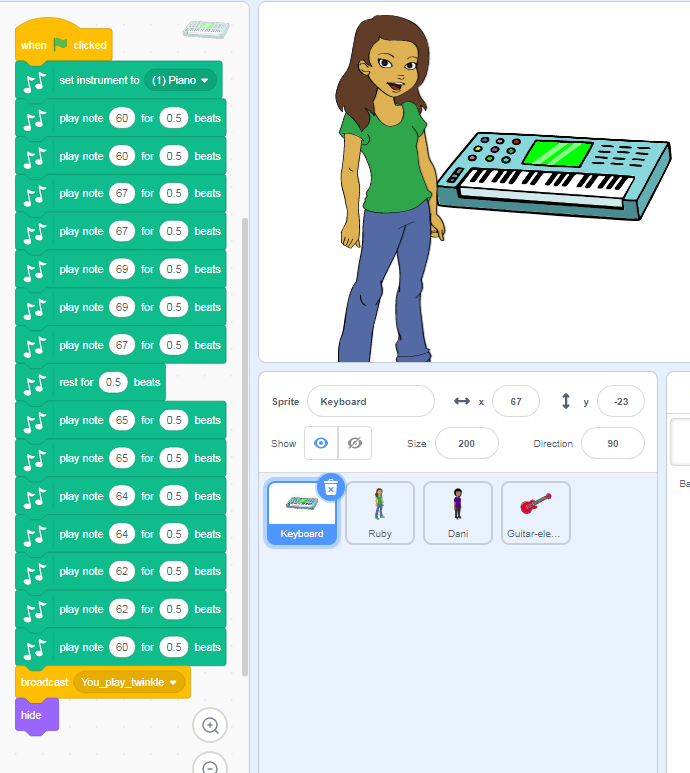


Image: Screen capture of Scratch programming blocks, Melody 1.

Step 2.

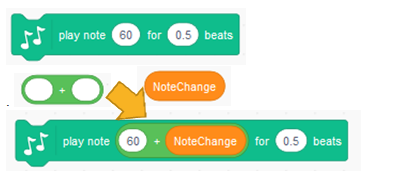
Make a second sprite that plays the same melody as Sprite 1. This program will include the NoteChange and random block. The note change is applied to each note in the melody. The second sprite will start by including the ‘When I receive message’ block. The show button is used to show the sprite on screen.

To mimic AI, use a random number generator to generate a number between [–16, 16] so that each tone gets played lower or higher (–16 represents two octaves lower, 16 represents two octaves higher). You could use other values [–8, 8] or, even simpler, [–2, 2].



Create a variable called NoteChange. The variable **NoteChange** stores the random number. NoteChange is added to each note in the melody. Use the **operator** block to add the original note and the NoteChange variable; for example, the C note (60) played for 0.5 beats.

.



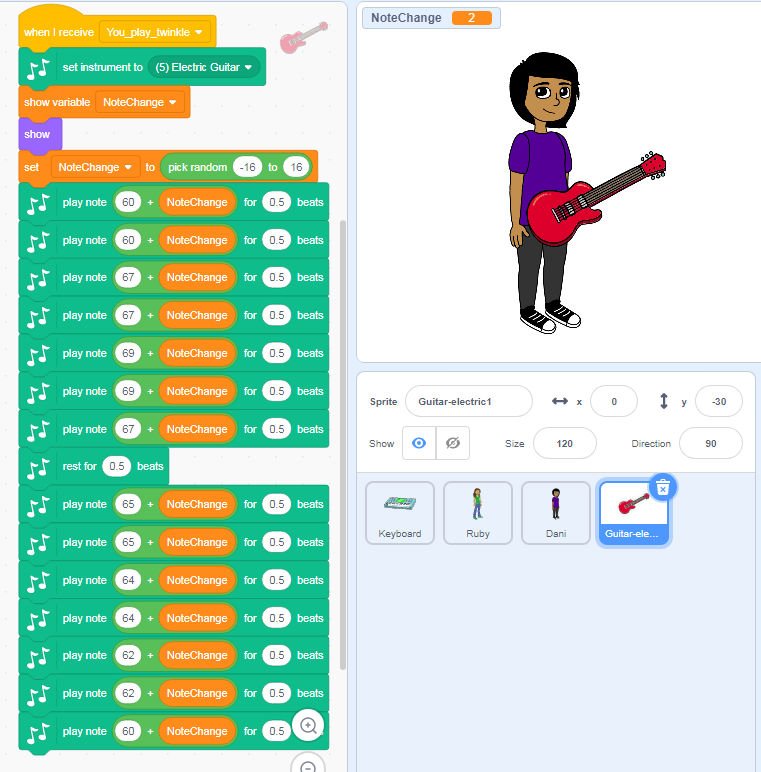


Image: Screen capture of Scratch programming blocks, response to Melody 1.

Students can view the code in the sample program and remix the project. In this code sample, the guitar player responds only once.

**Ideas for a remix**

Students:

* change the value of the random numbers generated (instead of –16, 16)
* change the value of the notes or beat
* change the instrument type
* add further programming blocks so that the guitar broadcasts a message for keyboard to play in response.
* remove the original notes sequence and replace with their own notes to create their own melody
* create the program, based on their own design of the algorithm and user interface.

Sample code: [Duet](https://scratch.mit.edu/projects/336284349/)

**Level Six: Program a duet between two instruments, adding beat changes   
(skill level: medium to high)**

This will provide a further challenge. Some students may want to explore how to make the musicians respond to each other with changing tone and beat. To do this as shown here in [Scratch Duet note and beat changes](https://scratch.mit.edu/projects/337259495/), they will need to create a variable for changing notes and changing beat. Add backdrops for each musician. The beat can be multiplied or divided by the random number. For example:

* a beat of 0.5 divided by random number [2] will change the beat to 0.25 and result in the melody being played twice as fast as the melody when using a 0.5 beat.

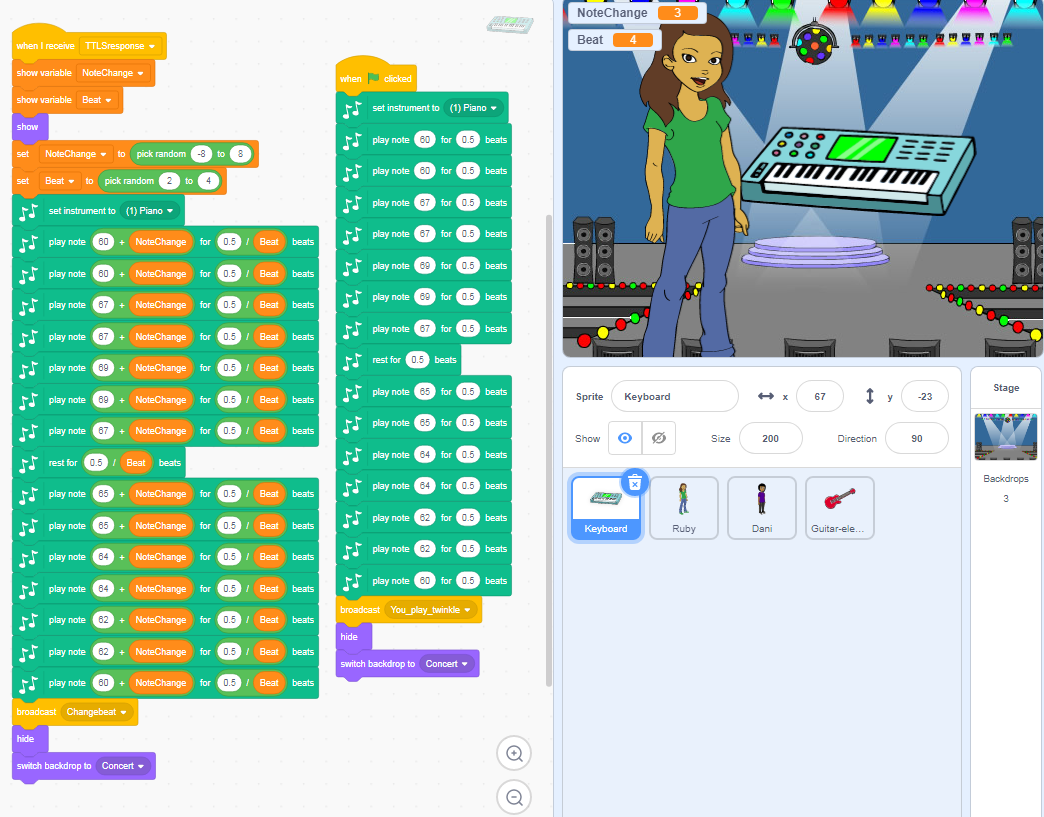


Image: Screen capture of Scratch programming blocks changing tone and beat.

**Ideas for a remix**

Students:

* change the value of the random numbers generated (instead of –16, 16)
* change the value of the notes or beat
* change the instrument type
* remove the original notes sequence and replace with their own notes to create their own melody
* create the program, based on their own design of the algorithm and user interface.

Sample code: Scratch [Duet note and beat changes](https://scratch.mit.edu/projects/337259495/)

# Discussion

Share what you have learned about AI and how ‘smart’ a computer can be. Compare the use of the random function used in the Visual Programming to mimic AI.

* Could the program be improved to better mimic AI?

# Why is this relevant?

Algorithms and programming are essential to developing machines powered by artificial intelligence (AI). AI is the ability of machines to mimic human capabilities in a way that we would consider 'smart'.

In conventional programming the computer is provided with a set of instructions for a defined set of scenarios. In the music-making programs we have used above, the students hard coded the program with specific inputs to create an output. To mimic the AI we used the random function to come up with a random response to the first played instrument.

Machine learning (ML) is an application of AI. With machine learning, we give the machine lots of examples of data, demonstrating what we would like it to do so that it can figure out how to achieve a goal on its own. The machine learns and adapts its strategy to achieve this goal.

In the AI Duet example, the machine would have been fed enormous amounts of data on musical notes and rhythms. The AI has learned to respond to a sequence of notes that work in harmony with the input.

This lesson focuses on:

* representation of music
* the difference between explicit programming and learning by example as in AI.

# Resources

* [Fostering Computational Thinking in Primary School through a LEGO-based Music Notation](https://www.researchgate.net/figure/A-LEGO-based-representation-of-rhythm-in-use-at-the-Lego-Music-center_fig1_319445103)
* AI powered computer tool [AI Duet](https://experiments.withgoogle.com/ai/ai-duet/view/)
* [Scratch 3.0](https://scratch.mit.edu)

# Assessment

## Teacher assessment

Choose from these suggested assessment approaches and tasks those that will best suit your students.

|  |  |
| --- | --- |
| **Possible tasks** | **Relevant content descriptor(s)** |
| Represent notes of a sequence that you used in your programming. | (ACTDIK008)/ (ACTDIK015) |
| Describe the hardware and software used in making your musical instrument or duet. Explain how it works. What data is used and stored?  How did you design and create your user interface? | ACTDIK007 / ACTDIK014  (ACTDIP010) /( ACTDIP018)  ACTDIP019/ ACTDIP020 |
| Draw a labelled diagram of the AI Duet in action. Identify (label) your inputs and the AI outputs.   * Describe how the AI responds to different keys and rhythms. * Compare the AI to your programmed musical instrument or Duet. * Explain how your solution meets the needs of the user. | [(ACTDIK014)](http://www.australiancurriculum.edu.au/curriculum/contentdescription/ACTDIK001) / [(ACTDIK023)](http://www.scootle.edu.au/ec/search?accContentId=ACTDIK007)  (ACTDIP012) /(ACTDIP021) |
| How might this type of AI be used in our daily lives?  This AI is a ‘recommender’ type of AI. It plays the notes and melody in response to what the human played. It basically plays notes it thinks you will like to hear.  Netflix uses a recommender-type system. What similar uses can you think of for this kind of technology? | (ACTDIP021)/(ACTDIP031) |

# Australian Curriculum Alignment

## Technologies – Digital Technologies

**Years 3–4**

Identify and explore a range of digital systems with peripheral devices for different purposes, and transmit different types of data (ACTDIK007)

Recognise different types of data and explore how the same data can be represented in different ways (ACTDIK008)

Define simple problems, and describe and follow a sequence of steps and decisions (algorithms) needed to solve them (ACTDIP010)

Implement simple digital solutions as visual programs with algorithms involving branching (decisions) and user input (ACTDIP011)

Explain how student solutions and existing information systems meet common personal, school or community needs (ACTDIP012)

**Years 5–6**

Examine the main components of common digital systems and how they may connect together to form networks to transmit data (ACTDIK014)

Examine how whole numbers are used to represent all data in digital systems (ACTDIK015)

Design a user interface for a digital system (ACTDIP018)

Design, modify and follow simple algorithms involving sequences of steps, branching, and iteration (repetition) (ACTDIP019)

Implement digital solutions as simple visual programs involving branching, iteration (repetition), and user input (ACTDIP020)

Explain how student solutions and existing information systems are sustainable and meet current and future local community needs (ACTDIP021)

## The Arts: Music

**Years 3–4**

Develop aural skills by exploring, imitating and recognising elements of music including dynamics, pitch and rhythm patterns (ACAMUM084)

Practise singing, playing instruments and improvising music, using elements of music including rhythm, pitch, dynamics and form in a range of pieces, including in music from the local community (ACAMUM085)

**Years 5–6**

Explore dynamics and expression, using aural skills to identify and perform rhythm and pitch patterns (ACAMUM088)

Develop technical and expressive skills in singing and playing instruments with understanding of rhythm, pitch and form in a range of pieces, including in music from the community (ACAMUM089 - Scootle)