# General purpose programming

**Year levels**: 7–8

Please refer to the online lesson plan on the DT Hub to access all website links and additional resources.

Investigate home automation systems, including those powered by artificial intelligence (AI) with speech recognition capability.

These suggested activities for year levels 7-8 are designed for students using general purpose programming languages JavaScript and Python, with similar content to the visual coding lesson Home automation programming for year levels 5-6.

Note, the programming activities assume some previous experience with either JavaScript or Python, utilising branching (decisions), iteration (loops) and functions. This lesson is not suitable as a first introduction to these languages (consider the Visual to Text Coding lessons first).

This set of lesson ideas is related to Home automation with AI, which explores how AI works within the context of home automation and raises topics including risks and also how these systems might assist people living with disability.

# Before you begin

Before using any new technology tools undertake a risk assessment using the eSafety Commissioner’s New Technology: Risk assessment.

The use of speech recognition with Python and JavaScript brings specific hardware and software requirements. See the instructions below. (Note, the same activities are provided for both languages.)

## JavaScript

Students will use online environment Pencil Code for speech recognition.

|  |  |
| --- | --- |
| Warning | **Computer / laptop with microphone** required. |
| Warning | **Chrome browser** required. |
| Information | Pencil Code launches with a less common language CoffeeScript by default. Switch to JavaScript using the cog button  above the code area. Blocks for JavaScript also become available once all CoffeeScript code is removed. |

## Python

Speech recognition is not currently available in an online environment for Python.

|  |  |
| --- | --- |
| Warning | **Computer / laptop** required. |
| Warning | **Microphone use** is not available. Students must pre-record audio clips of their vocal commands (eg. using a tool like *Audacity*) and save these in the folder with their Python programs. |
| Warning | **Software installation** required. We recommend *Thonny*, a simple IDE that comes with Python built in and makes additional modules easy to install.   1. Download and run the *Thonny* installer from thonny.org 2. Start *Thonny*. 3. Click **Tools** → **Manage packages…** 4. In the search bar at top, type speechrecognition, then click **Find package from PyPI**. 5. Click **Install** to add the Speech Recognition module to *Thonny*. 6. In the search bar at top, type playsound, then click **Find package from PyPI**. 7. Click **Install** to add the playsound module to *Thonny*. 8. Click **Close** to return to the *Thonny* main screen. |
| Information | *Optional: To investigate the use of microphone in Python, try realpython.com.*  *The compatibility of audio modules for microphone use varies by Python version and microphone use requires installation of additional software on Windows and other platforms.* |

# Test your setup

## JavaScript

To test speech recognition at Pencil Code:

1. Visit dthesa.pencilcode.net/edit/speechTest to load a simple JavaScript program.
2. Press **play** to run the program. It should record something you say, then say it back to you.

*If the program does not work:*

* *Check the hardware and software requirements under* ***Before you begin****.*
* *Adjust the microphone or speaker volume on your computer.*
* *Disable any other software that might be using your microphone, eg. a web chat.*

### Privacy warning!

Avoid students recording their own voice in the pencil code environment. Generally an AI speech recognition system records the voice to then compare to its database of known speech patterns and data. The recordings would be stored on a server. This digital file may not be deleted.

To mitigate risk of a student’s voice being saved, use a device that has text to speech functionality. For example open an app on your tablet device or in your browser such as https://www.naturalreaders.com/online/. Type text in and play speech. The pencil code should operate based on this speech input.

## Python

To test speech recognition in *Thonny*:

1. Ensure you have installed *Thonny* and additional modules as per the instructions under **Before you begin**.
2. Download 'testing123.wav', an audio clip with voice saying "Testing 123".
3. Download 'speech\_test.py', a python program, and place it in the same folder as 'testing123.wav'.
4. Start *Thonny* and open the program 'speech\_test.py'.
5. Run the program.

*If the program does not work:*

* *Check the hardware and software requirements under* ***Before you begin****.*
* *The folder with the two files may be buried deep in a folder hierarchy with long names. Try moving the files to a simpler folder before opening again in* Thonny*.*
* *Adjust the speaker volume on your computer.*

# Suggested steps

**Unplugged activity**

Discuss home automation. Ask, ‘How might voice commands be used in home automation?’ Consider aspects of home automation such as:

* security and access
* climate control
* lighting
* entertainment.

Discuss the role of AI in performing the automation. This may prompt students to mention the use of personal assistants.

Ask students to consider how home automation can or might assist those living with disability.

With new technologies there are often risks to consider. Ask what could go wrong. Discuss risks such as privacy breaches from a system being hacked or times when the system might be down due to a power outage (including one caused by a natural disaster such as fire or flood).

**Plugged activity**

**Step 1: Activating and deactivating a light**

Use these suggested activities to explore implementing a digital solution that demonstrates how to control appliances, and to further investigate home automation.

**Use binary numbers 0 and 1 to switch a light on and off (skill level: easy)**

Use JavaScript or Python (without any form of AI) to create a program that switches appliances on and off using an input: 0 or 1. This task demonstrates the use of binary digits to change state; for example, ‘On to off’ or ‘Off to on’. As this program does not use any form of AI, use it to demonstrate how conventional coding is used to hard-code a particular action.

Depending on students’ familiarity with and understanding of binary numbers, review the use of 0 and 1 and explain that binary means ‘two states’. The two states can be represented in different ways, such as by the numbers 1 and 0, or by text (‘true’ and ‘false’, or ‘on’ and ‘off’). The main point here is that a binary device can be in just one of two possible states. A binary bit is a single on/off value.

**Abstraction and representation**

Abstraction is all about focusing on important information and excluding non-essential information. In this example of an electrical toggle switch, features such as size, colour, and placement are not essential for our purpose. The important information is that it operates in two states: on and off. The switch has many different representations: binary digits (1/0), text (on/off), true/false, or visually, as an electrical symbol. Here is an example of that, showing ‘Switch off’:

## Suggested steps - JavaScript

Follow the steps below to implement the program at Pencil Code. (Be sure to read **Before you begin** and **Test your setup** first.)  
See the online lesson plan for the completed Step 1 program.

1. Start a new JavaScript program at Pencil Code. Use this shortcut to an empty JavaScript program, or follow these steps:
   1. Go to pencilcode.net.
   2. Click **Let's Play!**
   3. Click the cog button  above the code area and select JavaScript.
   4. Delete any existing CoffeeScript code.
2. Write a program to show a light going on and off when the user types "0" / "on" or "1" / "off". This can be done with two simple functions activated by **keydown** events, which change the image for a sprite.

// Create and position the sprite.

var light = new Sprite();

light.jumpto(-150, 0);

// Show the light on.

function showLightOn() {

light.wear('http://www.digitaltechnologieshub.edu.au/docs/default-source/ai-lessons/general-purpose-programming\_year-78/light-on.svg');

}

// Show the light off.

function showLightOff() {

light.wear('http://www.digitaltechnologieshub.edu.au/docs/default-source/ai-lessons/general-purpose-programming\_year-78/light-off.svg');

}

// Main program.

light.wear('http://www.digitaltechnologieshub.edu.au/docs/default-source/ai-lessons/general-purpose-programming\_year-78/light-off.svg');

write('Press 1 to turn the light on, or 0 to turn it off.');

keydown('1', showLightOn);

keydown('0', showLightOff);

## Suggested steps - Python

Follow the steps below to implement the program in *Thonny*. (Be sure to read **Before you begin** and **Test your setup** first.)

1. Start a new Python program in *Thonny*. Save the file with a relevant file name, for example light\_switch\_text\_input.
2. Write a program to demonstrate a light going on and off when the user types "0" / "on" or "1" / "off". This can be done with a simple loop that runs forever. (Use *Thonny's* STOP button to end the program.)

print('Type 1 to turn the light on or 0 to turn it off.')

**while True:**

command = input()

**if** (command == '1'):

print('The light is ON.')

**if** (command == '0'):

print('The light is OFF.')

## Tinker idea

* Try adding a second appliance with a different input command to turn it on or off.

**Step 2: Adding speech recognition**

Present the challenge of creating a program that recognises the students’ voice commands to perform an action such as turning home appliances on or off.

Discuss the types of voice commands that might typically be used, for example:

Turn on/turn off

Lights on/Lights off

Turn fan on/Turn fan off.

Emphasise short commands that are easy to understand.

## Suggested steps - JavaScript

Follow the steps below to implement the program at Pencil Code. (Be sure to read **Before you begin** and **Test your setup** first.)  
Refer to the online lesson plan for the completed Step 2 program.

1. Begin with the completed Step 1 JavaScript program at Pencil Code.
2. Modify the program so that:
   * a single function is triggered by **listen**,
   * the function checks the recognised speech text for keywords,
   * it changes the sprite based on whether keyword 'on' or 'off' is found,
   * finally, it sets up the **listen** trigger again.

// Create and position the sprite.

var light = new Sprite();

light.jumpto(-150, 0);

// Change the light based on speech text.

function changeLight(*speechText*) {

if (speechText.includes('on')) {

light.wear('http://www.digitaltechnologieshub.edu.au/docs/default-source/ai-lessons/general-purpose-programming\_year-78/light-on.svg');

}

if (speechText.includes('off')) {

light.wear('http://www.digitaltechnologieshub.edu.au/docs/default-source/ai-lessons/general-purpose-programming\_year-78/light-off.svg');

}

listen(changeLight);

}

// Main program.

light.wear('http://www.digitaltechnologieshub.edu.au/docs/default-source/ai-lessons/general-purpose-programming\_year-78/light-on.svg');

write('Speak to turn the light on and off.');

listen(changeLight);

## Suggested steps - Python

Follow the steps below to implement the program in *Thonny*. (Be sure to read **Before you begin** and **Test your setup** first.)

Refer to the online lesson plan for the completed Step 2 program.

1. Have students create audio files with vocal commands like 'Light switch on' and 'Light switch off'. These should be saved as WAV files in the same folder as your Python programs. (Or download premade audio files: On, Off).
2. In *Thonny*, start with the completed Step 1 Python program and save with a fresh filename.
3. Modify the program to include the speech recognition module. We'll still have the user type 1 or 0, but now it will activate a function to load and play the relevant audio clip then try to recognise the voice command.

**import** speech\_recognition

**import** playsound

# Play audio clip and attempt to recognise speech.

**def** **recogniseCommand**(filename):

print('Now testing', filename)

playsound.playsound(filename)

recognizer = speech\_recognition.Recognizer()

source = speech\_recognition.AudioFile(filename)

audio = recognizer.listen(source)

text = recognizer.recognize\_google(audio)

print('I heard:', text)

**if** 'on' in text:

print('The light is ON.')

**elif** 'off' in text:

print('The light is OFF.')

**else**:

print('Voice command unrecognised.')

# Main program.

print('Type 1 to turn the light on or 0 to turn it off.')

**while True**:

command = input()

**if** (command == '1'):

recogniseCommand ('lightSwitchOn.wav') # Use your audio clip filename here.

**if** (command == '0'):

recogniseCommand ('lightSwitchOff.wav') # Use your audio clip filename here.

## Tinker idea

* Add an activation phrase like "OK, House" or "Hey, House". Commands to turn the light on or off will be ignored until this phrase is first recognised.
  + For JavaScript:
    - Refer to the online lesson plan for a completed solution to this tinker idea.
  + For Python:
    - Download a premade audio file for "OK, House".
    - Refer to the online lesson plan for a completed solution to this tinker idea.

**Challenge: Multiple appliances**

Create a program that recognises the students’ voice commands to turn multiple home appliances on or off.

Discuss the types of voice commands that might typically be used, for example:

* Turn on/turn off
* Lights on/Lights off
* Turn radio on/Turn radio off.

Emphasise short commands that are easy to understand, or keywords to scan for.

## Sample solution - Javascript

Download premade .svg files: "Radio on", "Radio off"

Refer to the online lesson plan for a completed sample solution.

## Sample solution - Python

Download premade audio files: "Radio on", "Radio off"

Refer to the online lesson plan for a completed sample solution.

# Discussion

Share what you have learned about AI and how ‘smart’ a computer can be. Think about the program you created:

* What were some of the challenges?
* Where the program used AI, what things affected how well it worked?
* How might you improve the program if you did the task again?

# 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 binary program, the students hard-coded the program with specific inputs of 0 or 1 to turn the appliance off/on. The next program included speech recognition.

* In the JavaScript program the user's speech was detected via a microphone (input) and this command was recognised using a form of AI, Natural Language Processing (NPL).
* In the Python program the user's speech was pre-recorded as a WAV file, which provided the input. The program recognised the command using a form of AI, Natural Language Processing (NPL).

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 our example there are many variations on commands to turn appliances on and off. The use of short clear and precise commands increases the accuracy of the AI speech recognition.

This lesson focuses on:

* binary numbers (1, 0)
* speech recognition

# Resources

Online JavaScript and CoffeeScript programming environment Pencil Code

Offline Python programming environment Thonny

# Assessment

## Teacher assessment

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

|  |  |
| --- | --- |
| **Possible tasks** | **Relevant content descriptor(s)** |
| Investigate the hardware and software involved to capture and store audio in digital systems. Describe the steps in the process that allows vibrations in the air to be represented as a sequence of binary digits.  Working as a team, choose a demographic for your intended audience and present your findings in an online format. | (AC9TDI8K04) / (AC9TDI8P11) / (AC9TDI8P12) |
| Investigate the speech recognition system, ie. the software that takes digitised sound and recognises the speech to produce text. What is the role of **phonemes** and **machine learning**? Why does your device need to be connected to the Internet for speech recognition to work?  (This page section gives a brief overview of modern speech recognition techniques.)  Working as a team, choose a demographic for your intended audience and present your findings in an online format. | (AC9TDI8K04) / (AC9TDI8P11) / (AC9TDI8P12) |
| Create a flowchart, written procedure or pseudocode to explain the algorithm design for the Step 2 program. What data is used and stored? | (AC9TDI8P05) / (AC9TDI6P06) |
| Design a program to model an application that uses speech recognition for a purpose other than home automation (eg. accessibility, health, education, entertainment). Use a flowchart, written procedure or pseudocode to describe the algorithm(s).  Code and test the program in JavaScript or Python. | (AC9TDI8P04) / (AC9TDI6P07) / (AC9TDI8P08) / (AC9TDI8P09) / (AC9TDI8P05) / (AC9TDI6P06) |
| List some examples of real world applications of speech recognition.  What issues might arise with these applications (eg. ethical, safety, sustainability, social)? | (AC9TDI8P04) / (AC9TDI8P10) |

# Australian Curriculum Alignment

## Technologies – Digital Technologies

**Years 7–8**

* Explain how and why digital systems represent integers in binary (AC9TDI8K04)
* Select and use a range of digital tools efficiently, including unfamiliar features, to create, locate and communicate content, consistently applying common conventions (AC9TDI8P11)
* Select and use a range of digital tools efficiently and responsibly to share content online, and plan and manage individual and collaborative agile projects (AC9TDI8P12)
* Evaluate existing and student solutions against the design criteria and user stories and their broader community impact (AC9TDI6P06)
* Design algorithms involving nested control structures and represent them using flowcharts and pseudocode (AC9TDI8P05)
* Trace algorithms to predict output for a given input and to identify errors (AC9TDI8P06)
* Define and decompose real-world problems with design criteria and by creating user stories (AC9TDI8P04)
* Select and use appropriate digital tools effectively to create, locate and communicate content, applying common conventions (AC9TDI6P07)
* Generate, modify, communicate and evaluate alternative designs (AC9TDI8P08)
* Implement, modify and debug programs involving control structures and functions in a general-purpose programming language (AC9TDI8P09)
* Evaluate existing and student solutions against the design criteria, user stories and possible future impact (AC9TDI8P10)