Three random words

# Code snippets .

Code from the starter activity

|  | from random import randint  random\_number = randint(65,90)  random\_character = chr(random\_number)  print(random\_character) |
| --- | --- |

Create a new word that has replaced the letter a with an @ symbol

|  | word = "abc"  new\_word = ""  for letter in word:  if letter == "a":  new\_word = new\_word + "@"  else:  new\_word = new\_word + letter  print(new\_word) |
| --- | --- |

# Scenario .

A school technician would like a program that learners can use to generate a secure password. The program should:

* Prompt the user to enter three individual random words
* Each entered word should be replaced with **lower case letters**
* Concatenate the three random words into a password
* Replace each vowel in the password with a random character based on the conversion table below
* Display the secure password to the user for them to use

**Conversion table**

| **Vowel** | **ASCII code** | **ASCII character** |
| --- | --- | --- |
| a | Between 33 and 37 | Between ! and % |
| e | Between 38 and 42 | Between & and \* |
| i | Between 43 and 47 | Between + and / |
| o | Between 58 and 61 | Between : and = |
| u | Between 91 and 94 | Between [ and ^ |

**Tip**: Use the code snippets on page 1 to help you with this challenge.

Here is some example input and output for your program to see how it might work.

| **Example** |  |
| --- | --- |
| **Note:** Given the input you see in this sample interaction, this is the output your program should produce. | |
| A prompt is given for the user to enter the first word. | Enter your first word: |
| The user enters a word. | Tree |
| A prompt is given for the user to enter the second word. | Enter your second word: |
| The user enters a word. | Fish |
| A prompt is given for the user to enter the third word. | Enter your third word: |
| The user enters a word. | Monkey |
| The program concatenates the three words into one variable. Each letter is checked for a vowel. If it is a vowel, then randomisation is applied. The characters are added to a new string. |  |
| The secure password is displayed for the user. | tr(&f.shm;nk)y |

# Explorer tasks .

**Task 1**

Add an additional bit of security to your program. It should ask the user if their screen is secure before presenting them with the password.

**Task 2**

The first and last letters of the password need to be upper case. Adjust your program to meet these new requirements.

**Task 3**

All passwords must be 12 characters or over in length. Adjust your program so that it checks the length of the password and asks the user to re-enter their three words if it is too short.

**Task 4**

How efficient is your code? Could you introduce your own functions to reduce repetition? Take a look at your final solution and decide how it can be adapted.