The 4 characteristics of a strong password :

one lowercase letter; ONE UPPERCASE; 1 digit; at least 8characters

Abstraction

We want to develop an algorithm that will report the password strength (as developed in the CT challenge for the **for** loop lessons) BUT as often as the user wishes. A **while** loop must be used in your program.

Some of the key components of the problem are :

- The password will be represented by a string.
- Each character in the string will need to be examined to determine if it is uppercase, lowercase or a digit.
- The user must be allowed to make a decision whether to continue with another password or quit the program.

Writing your Thinking

Take 5 minutes to think about how you would tackle this problem.

- **Would you check the length first so you would not have go any further? What does this mean for assessing the overall strength?**
- If a character is lower case, how will you store that information? What will you call the variable?
- **How will you accumulate the strength of the string?**
- Within the **while** loop, what question do you ask the user and how do you store their answer?
  If they say yes or no, then how does your **while** condition respond?

Using Think-Pair-Share-Square (TPSS), go through how you and your partner were thinking about how to solve the problem.

Pseudo-Code

```
While the_user_wants_to_continue {
    #Initialise the conditions
    Set all 4 characteristics to False;
    Ask the user for their password;
    #Check each character of the string if it meets a strong characteristic
    If length of password >7 { long enough password = True}
    for i = 1 to length of password {
        if password(i) == uppercase {one upper case char = True }
        else if password(i) == lowercase { one lower case char = True }
        else if password(i) == digit { one digit char = True }
    }
    Output the strength of the password from 0-4
    Ask the user if they want to continue.
    If they say NO { set the_user_wants_to_continue = False }
}
```

An extension of the challenge on the html page

Some Tkinter code can be viewed, using this html page, to set up a UI to ask the user for a string and output some information.

The Tkinter UI gives the user control over the number of times they can test the strength of a password.

Read the program and learn how it gives control to the user using a **submit** button and the **click** definition.

Can you enhance the Tkinter program to test the strength of a password and perhaps add some extra features?