**Passwords**

A network technician has set the following rules for passwords on their network:

* It has at least 8 characters
* It contains both upper case and lower case letters
* It contains at least one numeric digit
* It contains at least one symbol

So, a password like pyth0n would not be considered safe, whereas Pyth0n!stAs would.

Task 1 . Is it safe?

Create a Python program that prompts the user to enter a password and displays whether or not the password follows the technician’s rules. .

|  |  |
| --- | --- |
| **Example** | |
| Note: Use this example to check your program. This is the output your program should produce for the given input. | |
| The program displays a prompt and waits for keyboard input | Enter a password: |
| The user types in a password | pyth0n |
| The program displays a message that the password is not safe | pyth0n is not considered a safe password: |

|  |  |
| --- | --- |
| **Example** | |
| Note: Use this example to check your program. This is the output your program should produce for the given input. | |
| The program displays a prompt and waits for keyboard input | Enter a password: |
| The user types in a password | Pyth0n!stAs |
| The program displays a message that the password is safe | Pyth0n!stAs is considered a safe password. |

**Checklist**: Tick (✔) the corresponding box if your program:

|  |  |  |
| --- | --- | --- |
| |  | | --- | |  | | Prompts the user for a password to check. |
| |  | | --- | |  | | Computes the length of the password. |
| |  | | --- | |  | | Computes whether or not the password contains any lower case characters OR computes *how many* lower case letters the password contains. |
| |  | | --- | |  | | Computes whether or not the password contains any upper case characters OR computes *how many* upper case letters the password contains. |
| |  | | --- | |  | | Computes whether or not the password contains any digits OR computes *how many* digits the password contains. |
| |  | | --- | |  | | Computes whether or not the password contains any symbols OR computes *how many* symbols the password contains. |
| |  | | --- | |  | | Displays a message informing the user whether or not the password provided meets the network criteria. |

Task 2 . Why not?

**Extend** the program so that it displays the safety criteria that are not met, in case an ‘unsafe’ password is provided.

|  |  |
| --- | --- |
| **Example** | |
| Note: Use this example to check your program. This is the output your program should produce for the given input. | |
| The program displays a prompt and waits for keyboard input | Enter a password: |
| The user types in a password | pyth0n |
| The program displays the safety criteria that the password does not meet | Less than 8 characters  No uppercase letters  No symbols |

**Checklist**: Tick (✔) the corresponding box if your program:

|  |  |  |
| --- | --- | --- |
| |  | | --- | |  | | Displays the individual safety criteria that are not met by the provided password (see example). |

Explorer task . Generate a safe password

Create a program that generates a random password which meets the safety criteria outlined in the introduction.

Clues . Look here if you need help

**What are the variables I will need?**

Think about the quantities you will need to refer to in your program, i.e. the values that your program will need to keep track of.

You will probably need: the password entered by the user and a variable for its length.

To determine whether or not a password meets individual criteria, you could use variables such as lowercase\_count, uppercase\_count, digit\_count, and symbol\_count, to count the number of lower case letters, upper case letters, digits, and symbols in the password.

**How do I iterate over the characters in the password?**

Read the password as a piece of text. Use a for-loop to iterate over each character in the password. The **pseudocode** below illustrates the idea:

|  |  |
| --- | --- |
|  | for character in password:  process the character |

**How do I count the number of characters that... ?**

For any quantity that you will need to count, you will need a *counter* variable to keep track of that quantity. Every time you need to increase the counter, use an assignment like the one in the pseudocode below:

|  |  |
| --- | --- |
|  | *counter* = *counter* + 1 |

This statement can be read as ‘increase the counter by 1’. Don’t forget to initialise each counter to zero.

**How do I check if a character is... ?**

To check if a character belongs to a particular family of characters, use the in-operator:

|  |  |
| --- | --- |
|  | *character* in *text* |

The strings below will come in handy (copy and paste them in your code), for checking where each individual character in the password belongs:

|  |  |
| --- | --- |
|  | lowercase = 'abcdefghijklmnopqrstuvwxyz'  uppercase = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'  digits = '0123456789'  symbols = '!"#$%&\'()\*+,-./:;<=>?@[\\]^\_`{|}~' |

**How do I select a random character from a string?**

Import and use the choice function from the random module:

|  |  |
| --- | --- |
|  | from random import choice  *...*  *random character* = choice(*string*) |

**How do I assemble the random characters together in a string?**

Collect all the random characters into a list, say characters. Don’t forget to initialise the list of characters.

At the end of your program, use the code below to join all the characters you have collected in characters into a single password string:

|  |  |
| --- | --- |
|  | password = "".join(characters) |

**How do I randomly shuffle the items of a list?**

Depending on how you develop your solution, you might want to randomly shuffle the list of characters you generate for your password. In order to do that, import and use the shuffle function from the random module:

|  |  |
| --- | --- |
|  | from random import shuffle  *...*  shuffle(*list*) |

Resources are updated regularly — the latest version is available at: [ncce.io/tcc](http://ncce.io/tcc).

This resource is licensed under the Open Government Licence, version 3. For more information on this licence, see [ncce.io/ogl](http://ncce.io/ogl).