Projects

Password Generator

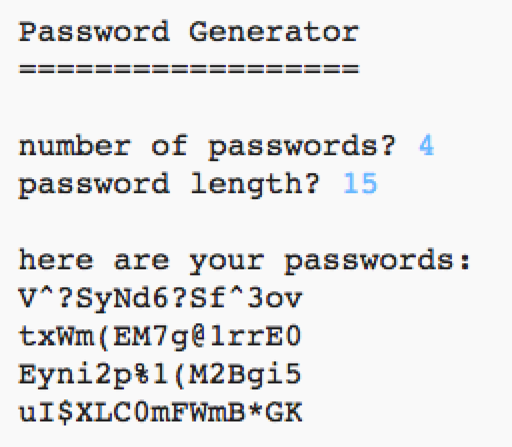
Create your own passwords that no one will be able to guess!

Python

Step 1 Introduction:

It’s important to protect your personal information online, and in this project you’ll create a program to generate passwords for you.

The passwords will be random, so no one will be able to guess them!



```
Password Generator
==================

number of passwords? 4
password length? 15

here are your passwords:
V^?SyNd6?Sf^3ov
txWm(EM7g@1rrE0
Eyni2p%1(M2Bgi5
uI$XLC0mFWmB*GK
```

Additional information for club leaders

If you need to print this project, please use the **Printer friendly version** (https://projects.raspberrypi.org/en/projects/password-generator/print).

Club leader notes

Introduction:

In this project, children will learn what makes a good password, and how to make a program that creates randomly generated passwords.

This project has been written for Safer Internet Day 2017, which is on 7th February 2017. The aim of Safer Internet Day is to promote the safe and responsible use of technology for young people. For more information visit **saferinternet.org.uk** (https://www.saferinternet.org.uk/) where you'll find an **education pack for 7-11 year-olds** (https://d1afx9quaogywf.cloudfront.net/cdn/farfuture/_-EgL7dYtxtypvvDcNCE53bYE-OMfdH59vaJ5XPcoG4/mtime:1483547665/sites/default/files/SID2017%20Education%20Pack%20for%207-11%20year%20olds_0.zip) containing additional resources.

Online Resources

This project uses Python 3. We recommend using **Trinket** (https://trinket.io/) to write Python online. This project contains the following Trinkets:

- **New (blank) Python Trinket – jumpto.cc/python-new** (http://jumpto.cc/python-new)

There is also a trinket containing the finished project:

- **'Password Creator' Finished – trinket.io/python/08c0ad3359** (https://trinket.io/python/08c0ad3359)

Offline Resources

This project can be **completed offline** (https://www.codeclubprojects.org/en-GB/https://projects-static.raspberrypi.org/projects/password-generator/1692c4a4698396c44ce1e8ecfee5a649f39522ab/en/resources/python-working-offline/) if preferred.

You can find the completed project in the 'Volunteer Resources' section, which contains:

- password-creator-finished/passwords.py

(All of the resources above are also downloadable as project and volunteer `.zip` files.)

Learning Objectives

- Repetition;
- The `random.choice()` method;

This project covers elements from the following strands of the **Raspberry Pi Digital Making Curriculum** (http://rpf.io/curriculum):

- **Combine programming constructs to solve a problem.** (https://www.raspberrypi.org/curriculum/programming/builder)

Challenges

- “Creating a better password” - using **howsecureismypassword.net** (https://howsecureismypassword.net/) to create secure passwords.
- “Using numbers and punctuation” - adding text to a string variable, giving a wider choice of random characters.
- “A longer password” - modifying the number of times a random character is chosen.
- “Choosing the number of passwords” - using a variable to specify the number of passwords required.

Project materials

Project resources

- **.zip file containing all project resources** (https://projects-static.raspberrypi.org/projects/password-generator/1692c4a4698396c44ce1e8ecfee5a649f39522ab/en/resources/password-generator-resources.zip)
- **Online blank Python Trinket** (http://jumpto.cc/python-new)
- **Offline blank Python file** (https://projects-static.raspberrypi.org/projects/password-generator/1692c4a4698396c44ce1e8ecfee5a649f39522ab/en/resources/new-new.py)

Club leader resources

- **.zip file containing all completed project resources** (https://projects-static.raspberrypi.org/projects/password-generator/1692c4a4698396c44ce1e8ecfee5a649f39522ab/en/resources/password-generator-finished.zip)

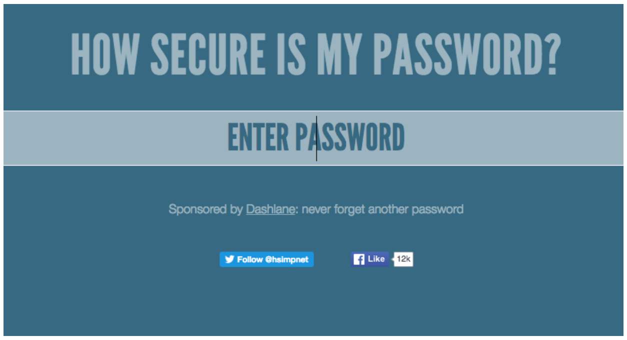
- **Online completed Trinket project** (https://trinket.io/python/08c0ad3359)
- **Offline completed project** (https://projects-static.raspberrypi.org/projects/password-generator/1692c4a4698396c44ce1e8ecfee5a649f39522ab/en/resources/password-generator-finished-passwords.py)

Step 2 How secure is your password?

A computer could try to guess your password by using 'brute force' – this means trying out lots of passwords until it guesses the right one.

Let's find out how long it would take a computer to guess your password.

- Go to **howsecureismypassword.net** (https://howsecureismypassword.net/), which is a website for finding out how secure your passwords are.

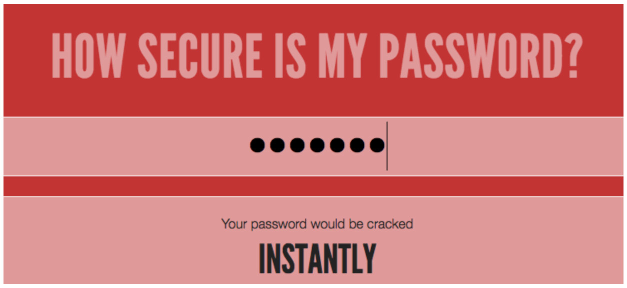


HOW SECURE IS MY PASSWORD?

ENTER PASSWORD

Sponsored by Dashlane: never forget another password

- Type in "letmein" (Let me in) as the password. You'll see that a computer would guess this password **instantly**!



HOW SECURE IS MY PASSWORD?

●●●●●●●

Your password would be cracked

INSTANTLY

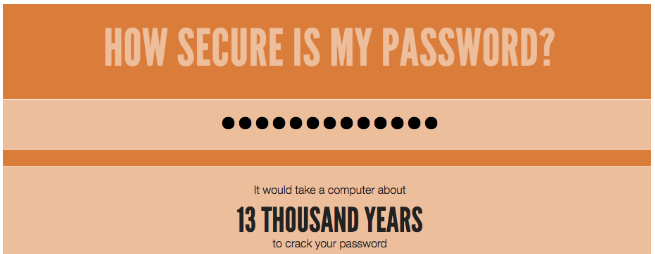
You'll also see some reasons why "letmein" isn't a good password to use:

- It's a very **common** password (one of the 15 most used passwords). A computer would guess these first.
- It contains words from the **dictionary**. A computer would also try these passwords first.
- It's very **short**. It would take a computer more time to guess a longer password.
- It only contains **letters**. Passwords are more secure if they also contain numbers and punctuation.

- Try entering a dictionary word. How long would it take a computer to guess that password?

Step 3 Challenge: Creating a better password

Can you enter a password that would take a computer more than 1,000 years to crack but isn't too long to type?



HOW SECURE IS MY PASSWORD?

●●●●●●●●●●●●●

It would take a computer about

13 THOUSAND YEARS

to crack your password

Remember that your password is harder to guess if it's:

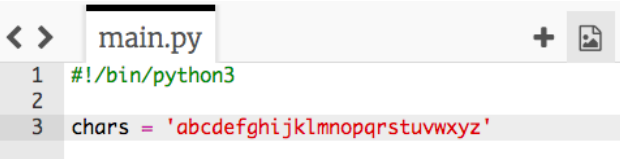
- Long
- Not a word in the dictionary
- Contains letters, numbers and punctuation

You're going to generate passwords that are hard for a computer to crack. These are useful for protecting important accounts. Note that many adults use a password manager program to help them remember lots of tricky passwords.

Step 4 Random characters

Let’s create a program to choose a random character for your password.

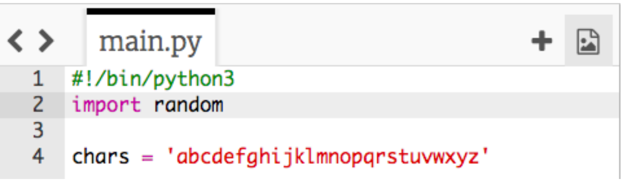
- Open the blank Python template Trinket: **jumpto.cc/python-new** (http://jumpto.cc/python-new).
- Create a list of characters, stored in a variable called `chars`.



```
#!/bin/python3

chars = 'abcdefghijklmnopqrstuvwxyz'
```

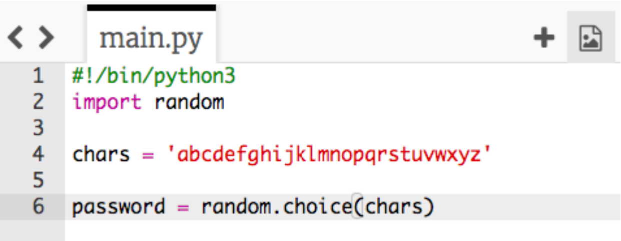
- To choose a random character, you’ll need to `import` the `random` module.



```
#!/bin/python3
import random

chars = 'abcdefghijklmnopqrstuvwxyz'
```

- Now you can choose a random character from the list, and store it in a variable called `password`.

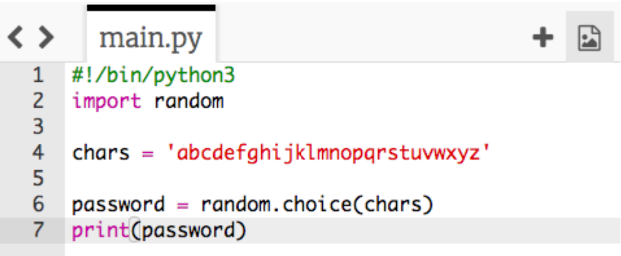


```
#!/bin/python3
import random

chars = 'abcdefghijklmnopqrstuvwxyz'

password = random.choice(chars)
```

- Finally, you can print your (very short!) password to the screen.

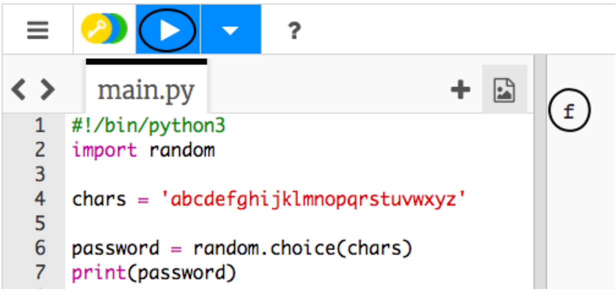


```
#!/bin/python3
import random

chars = 'abcdefghijklmnopqrstuvwxyz'

password = random.choice(chars)
print(password)
```

- Test your project by clicking 'run'. You should see a single random character on the screen.



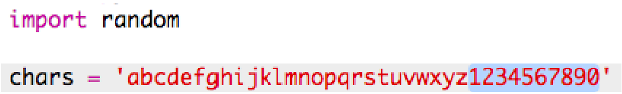
```
#!/bin/python3
import random

chars = 'abcdefghijklmnopqrstuvwxyz'

password = random.choice(chars)
print(password)
```

If you run your program a few times, you should see different characters appear.

- A password isn't very secure if it only contains letters. Add some numbers to your `chars` variable.



```
import random

chars = 'abcdefghijklmnopqrstuvwxyz1234567890'
```

- Test your code again a few times, and you should see that sometimes a number is chosen.

Step 5 Challenge: Using numbers and punctuation

Can you improve your program, so that it also chooses from:

- Capital letters (A-Z)
- Numbers (0-9)
- Punctuation (!?.,-)

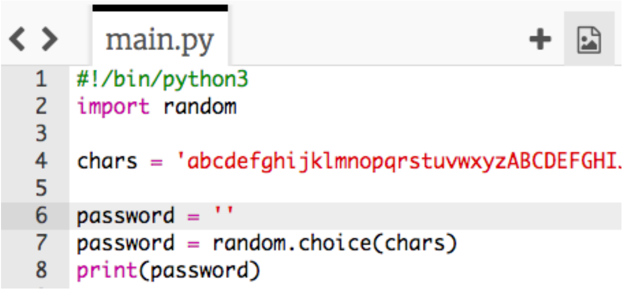
You'll need to add to your `chars` variable. Remember to test your improved program!

Step 6 A random password

A single character isn't very useful - let's improve your program to create a longer password.

- To create a password, you will add random characters to it, one at a time.

  To start with, your `password` variable should be empty. Add this line to your code:

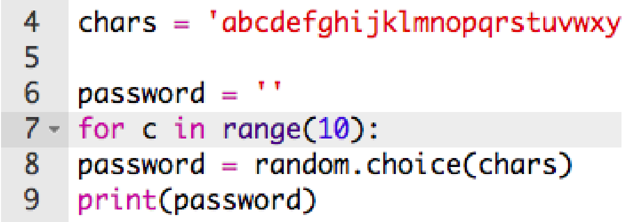


```
#!/bin/python3
import random

chars = 'abcdefghijklmnopqrstuvwxyzABCDEFGHI.

password = ''
password = random.choice(chars)
print(password)
```

- You want to choose a random character 10 times. To do this, add the following code:

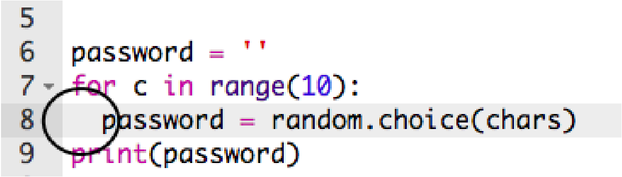


```
chars = 'abcdefghijklmnopqrstuvwxy

password = ''
for c in range(10):
password = random.choice(chars)
print(password)
```

- You should also indent (move in) the line to choose a random character, so that it happens 10 times.

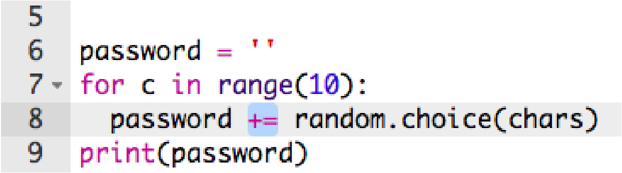
  To indent, press the 'tab' key.



```

password = ''
for c in range(10):
  password = random.choice(chars)
print(password)
```

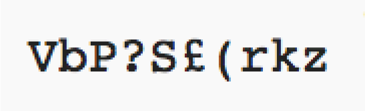
- You need to use `+=` to **add** the new character to the password each time.



```

password = ''
for c in range(10):
  password += random.choice(chars)
print(password)
```

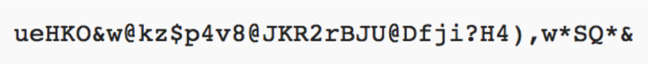
- Test your new code and you should see a password that’s 10 characters long.



```
VbP?S£(rkz
```

Step 7 Challenge: A longer password

Can you change your program so that it creates a verrrrrrrry long password?

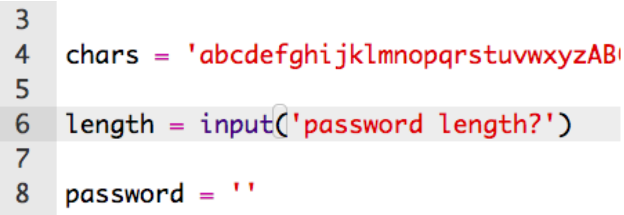


```
ueHKO&w@kz$p4v8@JKR2rBJU@Dfji?H4),w*SQ*&
```

Step 8 Choosing a password length

Some websites require passwords to be a certain length. Let’s allow the user to choose the length of their password.

- First, ask the user to input a password length, and store it in a variable called `length`.



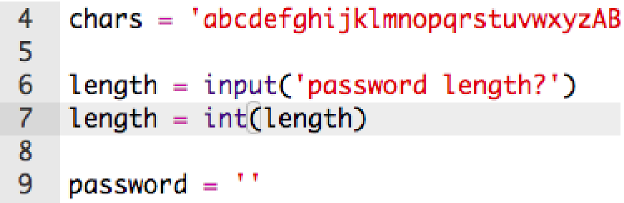
```

chars = 'abcdefghijklmnopqrstuvwxyzAB

length = input('password length?')

password = ''
```

- Use `int()` to turn the user’s input into a whole number.

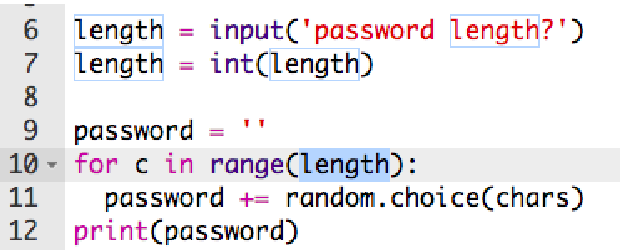


```
chars = 'abcdefghijklmnopqrstuvwxyzAB

length = input('password length?')
length = int(length)

password = ''
```

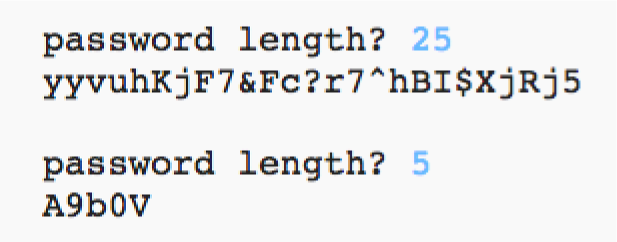
- Use your `length` variable to repeat as many times as the user entered.



```
length = input('password length?')
length = int(length)

password = ''
for c in range(length):
  password += random.choice(chars)
print(password)
```

- Test your code. The password created should be the length entered by the user.



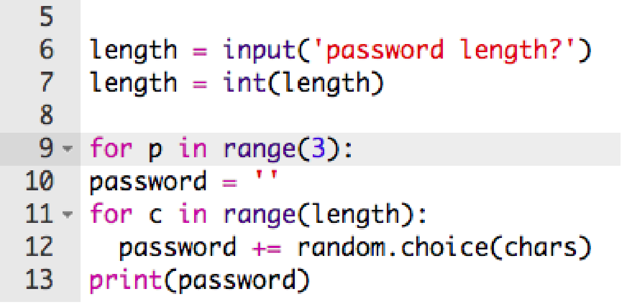
```
password length? 25
yyvuhKjF7&Fc?r7^hBI$XjRj5

password length? 5
A9b0V
```

Step 9 Lots of passwords

Let's allow the user to create 3 passwords at once.

- Add this code to create 3 passwords:

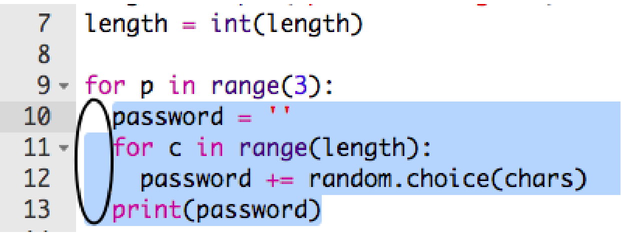


```

length = input('password length?')
length = int(length)

for p in range(3):
password = ''
for c in range(length):
  password += random.choice(chars)
print(password)
```

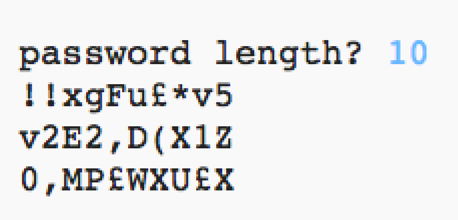
- Highlight the code for creating a password, and press tab to indent so that it repeats 3 times.



```
length = int(length)

for p in range(3):
  password = ''
  for c in range(length):
    password += random.choice(chars)
  print(password)
```

- Test your new code. You should now see 3 passwords of your chosen password length.

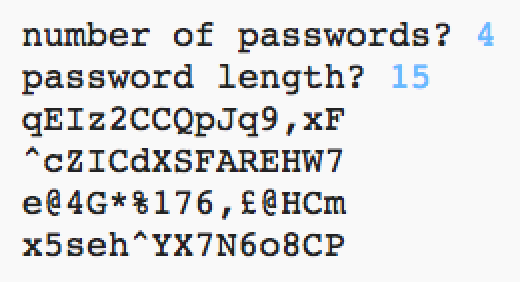


```
password length? 10
!!xgFu£*v5
v2E2,D(X1Z
0,MP£WXU£X
```

Step 10 Challenge: Choosing the number of passwords

Instead of always printing 3 passwords, can you allow the user to enter the number of passwords they want?

Here’s how your program should work:



```
number of passwords? 4
password length? 15
qEIz2CCQpJq9,xF
^cZICdXSFAREHW7
e@4G*%176,£@HCm
x5seh^YX7N6o8CP
```

The code you’ll need is **very** similar to the code for entering the `length` of the password.