Python

Hiding and encrypting passwords in Python?

maskpass()

```
pip install maskpass

import maskpass

pwd = maskpass.askpass(prompt="Password:", mask="#")

print(pwd)
```

Or

```
import maskpass

 pwd = maskpass.advpass()

print('Password : ', pwd)
```

base64()

The base64 encode and decode function both require a byte-like object. To convert a string into bytes, we must encode a string using Python’s built-in encode function

```
# Hiding the inputted password with maskpass()

# and encrypting it with use of base64()

import maskpass # to hide the password

import base64 # to encode and decode the password


# dictionary with username

# as key & password as value

dict = {'Rahul': b'cmFodWw=',

'Sandeep': b'U2FuZGVlcA=='}
```

```
# function to create password
def createpwd():
print("\n========Create Account=========")
name = input("Username : ")

# masking password with prompt msg 'Password :'
pwd = maskpass.askpass("Password : ")

# encoding the entered password
encpwd = base64.b64encode(pwd.encode("utf-8"))

# appending username and password in dict
dict[name] = encpwd
# print(dict)

# function for sign-in
def sign_in():
print("\n\n=========Login Page===========")
name = input("Username : ")

# masking password with prompt msg 'Password :'
pwd = maskpass.askpass("Password : ")

# encoding the entered password
encpwd = base64.b64encode(pwd.encode("utf-8"))

# fetching password with
# username as key in dict

password = dict[name]

if(encpwd == password):

print("Successfully logged in.")

else:

print("Login Failed")


# calling function

createpwd()

sign_in()
```

```
F:\files>python "userLogin.py"
========Create Account=========
Username : Rahulraj
Password : *****

=========Login Page===========
Username : Rahulraj
Password : *****
Successfully logged in.
```

How Can You Emulate Do-While Loops in Python?

Use a while Loop and the break Statement

```
while True:
    # Do some processing...
    # Update the condition...
    if condition:
        break
```

Infinitely loops ?

```
while True:
    print("I am always true")
```

Press `Control C` to escape and end the loop.

To apply this to a for loop

```
def infinity():
    while True:
        yield
```

This can be used as follows:

```
for _ in infinity():
    pass
```

```
import itertools

for _ in itertools.repeat([]):  # return an infinite iterator
    pass
```

```
for i, _ in enumerate(iter(bool, True)):
    input(i)
```
