EECS 565: Intro to Information and Computer Security
Homework 1

Bruteforce Attack Task 1:

```
# Task 1: The hash value of her password by MD5 is 94d9e03c11395841301a7aee967864ec.
# You know the password is a length of 14. (The password can be combined with A-Z, a-z, or 0-9)
def task1():
    target = "94d9e03c11395841301a7aee967864ec"
    p_len = 14
    chars = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"

    start_time = time.time()
    for combo in itertools.product(chars, repeat=p_len):
        password = ''.join(combo)
        if check(target, password, 1) == True:
            end_time = time.time() # Stop measuring time
            test_time = end_time - start_time
            print("Time to crack: ", test_time, "seconds")
```

Bruteforce Attack Task 2:

```
# Task 2: The hash value of her password by MD5 is f593def02f37f3a6d57bcbc9480a3316.
# You know the specific password format: <4-UPPER CASE LETTER><3-DIGIT-NUMBER><4-SMALLER CASE LETTER>
# e.g., XYEX234ascf
def task2():
    target = "f593def02f37f3a6d57bcbc9480a3316"
    upper = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
    lower = upper.lower()
    digits = '0123456789'

    start_time = time.time()
    for u in itertools.product(upper, repeat=4):
        for digit in itertools.product(digits, repeat=3):
            for l in itertools.product(lower, repeat=4):
                password = ''.join(u) + ''.join(digit) + ''.join(l)
                if check(target, password, 2) == True:
                    end_time = time.time()
                    test_time = end_time - start_time
                    return(print("Time to crack: ", test_time, "seconds"))
```

Bruteforce Attack Task 3:

```
# Task 3: The hash value of her password by MD5 is bfb2c12706757b8324368de6a365338b.
# You know the specific password format: the length is 11 and it includes 7 upper case letters
# and the number "1234". But you don't know the position of "1234"
def task3():
    target = "bfb2c12706757b8324368de6a365338b"
    p_len = 11
    upper = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
    digits = '1234'

    start_time = time.time()
    for u in itertools.combinations_with_replacement(upper, 7):
        for x in range(8):
            for digit in itertools.permutations(digits):
                password = ''.join(u[:x]) + ''.join(digit) + ''.join(u[x:])
                if check(target, password, 3) == True:
                    end_time = time.time()
                    test_time = end_time - start_time
                    return(print("Time to crack: ", test_time, "seconds"))
```

Check Function and Imports:

```python
import hashlib
import itertools
import time

def check(target, guessed_password, task): # Change the hash
    myhash = hashlib.md5(guessed_password.encode()).hexdigest()
    if target == myhash:
        print(f"Task {task}: Success! Password is: ", guessed_password)
        return True
    else:
        return False
```

Output:

```
Python 3.11.4 (v3.11.4:d2340ef257, Jun  6 2023, 19:15:51) [Clang 13.0.0 (clang-1
300.0.29.30)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: /Users/Kaitlyn/Library/Mobile Documents/com~apple~CloudDocs/KU/Curren
t Courses/EECS 565/Homework/HW 1/hw1.py
Task 2: Success! Password is:  AAAA123dddd
Time to crack:  29.526153326034546 seconds

Task 3: Success! Password is:  ABC1234GHIJ
Time to crack:  19.588798999786377 seconds
>>>
```

*Did not do output for task1 because it took too long

Task 1 took the longest amount of time. It had the longest password length and had any combination of all three groups of numbers, so it could take an extremely long time to find the correct combination.

Task 2 took the second longest amount of time. It has different combination of characters, but this time we know how many of each type of character will be present. We also know the format they will occur in.

Task 3 took the shortest amount of time. It was 11 characters and we knew that four characters in a row had to be 1234, which made it easier to guess and move around characters.