```python
# ITP 125 - Python Final Project
# Mantej Lamba, Prof Grant, Monday 6PM

"""
importing 3 modules into our program

first one is hashlib -- which will help us hash our passwords in the MD5 format and use
different functions in the
process to turn it into a md5 format

second one is time -- this helps us keep track of how long it took to generate the password and
then test it against the
text file

third one is itertools -- this helps us generate all possible permutations for our passwords so
we can hash them and check them against
the hashes.txt file
"""
import hashlib,time,itertools



# list of all possible characters that can be used to create the passwords
# Prof Grant mentioned in class that we could use this option to make our program faster and if
we didn't want to include all possible ASCII characters
password =  ["Z", "A", "D", "G", "o", "d", "1", "2", "3", "4", "b", "C", "E", "T", "r", "j",
"a", "n", "F", "g", "h", "t", "O", "P", "@", "s", "w"]

# creating a variable to store the time that we start the program
startTime = time.time()

# creating an empty list called lines
lines = []
# getting all of the lines from hashes.txt and storing them right after each other in the lines
list
# used .strip() to make sure that there were no leading or ending whitespaces that would have
messed up the hashing process
for line in open ("hashes.txt"):
    lines.append(line.strip())

# iterating through the length of the words so we can generate all the possible passwords for
each length
for i in range(8):
    # creating a length variable that is equal to the ith index of the for loop we are in
    length = i
    # finding all the possible permutations for the passwords of a specific length and turning
them them into a list of strings
    perm = list(map("".join, itertools.permutations(password,length)))
    # iterating through the list of permutations that we calculated just above
    for j in range(len(perm)):
        # encoding our password to convert it into bytes so it can be accepted by the hash
function
        encode = hashlib.md5(perm[j].encode())
        # turning our encoded variable, encode, into a hexadecimal format so we can compare it
to our encrypted password in our hashes.txt file
        hex = encode.hexdigest()
        # checking if hex is inside of the lines list
        if hex in lines:
            # if it is, then we take the time at this point
            endTime = time.time()
            # then we print whatever we got in the correct format (according to Blackboard),
and subtract the endTime from the startTime to get the total time it took
            print(perm[j], "took " + str(endTime - startTime) + " seconds to crack")
```