```python
import random
import os
import re
from math import gcd


def prime(primeList):
    a = random.randint(10000, 100000)
    p = random.randint(10000, 100000)

# only testing once as the probability of the same number twice is really low
    if(a == p):
        p = random.randint(10000, 100000)

    if(a > p):
        holder = p
        p = a
        a = holder

    if((p % 2) == 0):
        return

    exp = ((p - 1)/2)
    r = pow(int(a), int(exp), int(p))

    if(r == 1 or r == -1):
        primeList.append(p)

    return primeList


def egcd(a, b):
    if a == 0:
        return (b, 0, 1)
    else:
        z, x, y = egcd(b % a, a)
        return (z, y - (b // a) * x, x)


def calcD(e, n):
    varA, x, ignore = egcd(e, n)
    if varA == 1:
        return x % n


def encrypt(e, n, plaintext):
    cipher = pow(ord(plaintext), e, n)

    return cipher


def decrypt(d, n, ciphertext):
    decipher = chr(pow(ciphertext, d, n))

    return decipher

# The legit program, that Ill be using personally
def main():
    raw = {}
```

```
60      primes = []
61
62  # get primes
63      while(len(primes) != 2):
64          prime(primes)
65
66  # calcs for e, n ,d phi
67      n = primes[0] * primes[1]
68      phi = (primes[0] - 1) * (primes[1] - 1)
69
70      e = random.randrange(1, phi)
71      g = gcd(e, phi)
72
73      while g != 1:
74          e = random.randrange(1, phi)
75          g = gcd(e, phi)
76
77      d = calcD(e, phi)
78
79  # Writing out decryption keys
80      f = open("keys.txt", "a+")
81      f.write(" " + str(d) + "  " + str(n) + "\n")
82      f.close()
83
84      mode = input("Would you like to Encrypt: 1 or Decrypt: 2... : ")
85
86      inputFile = input("Please enter the name of the file: ")
87
88      if os.path.isfile(inputFile):
89          f = open(inputFile, "r", "\r\n")
90          inLines = f.readlines()
91
92          if(mode == "1"):
93              for i in range(0, len(inLines)):
94                  currentLine = list(inLines[i])
95
96                  f = open("encrypted.txt", "ab+", "\r\n")
97
98                  for j in range(0, len(currentLine)):
99                      ciphertext = encrypt(e, n, (currentLine[j]))
100
101 # It might seem odd that i have it print the value then two spaces
102 # I just did it that way because i already knew how to parse that
103                     f.write("%d  " % ciphertext)
104                 f.close()
105
106         if(mode == "2"):
107             d = int(input("Please Enter d: "))
108             n = int(input("Please Enter n: "))
109             for i in range(0, len(inLines)):
110                 currentLine = list(inLines[i])
111
112                 inLines[i] = re.sub(r"[ \t]{2,}", r", "
    inLines[i].rstrip())
113                 raw[i] = inLines[i].split(", ")
114
115                 parsed = list(raw.values())
116
117                 f = open("decrypted.txt", "wb", newline="\r\n")
118
```

```
                results = [int(i) for i in parsed[0]]

                for i in range(0, len(results)):
                    ciphertext = decrypt(int(d), int(n), results[i])

                    f.write(ciphertext)

            f.close()


# For the purpose of making testing easy
def assignment():
    os.system("rm -f encrypted.txt")
    os.system("rm -f decrypted.txt")
    raw = {}
    primes = []
# get primes
    while(len(primes) != 2):
        prime(primes)

# calcs for e, n ,d phi
    n = primes[0] * primes[1]
    phi = (primes[0] - 1) * (primes[1] - 1)

    e = random.randrange(1, phi)
    g = gcd(e, phi)

    while g != 1:
        e = random.randrange(1, phi)
        g = gcd(e, phi)

    d = calcD(e, phi)


# input("Please enter the name of the file: ")
    inputFile = "testfile-DES.txt"
    print("e is: ", e)
    print("d is: ", + d)
    print("n is: ", + n)

    if os.path.isfile(inputFile):
        f = open(inputFile, "r")
        inLines = f.readlines()

        for i in range(0, len(inLines)):
            currentLine = list(inLines[i])

            f = open("encrypted.txt", "a+",)

            for j in range(0, len(currentLine)):
                ciphertext = encrypt(e, n, (currentLine[j]))

# It might seem odd that i have it print the value then two spaces
# I just did it that way because i already knew how to parse that
                f.write("%d  " % ciphertext)
            f.close()

        f = open("encrypted.txt", "r")
        inLines = f.readlines()

```

```
        for i in range(0, len(inLines)):
            currentLine = list(inLines[i])

            inLines[i] = re.sub(r"[ \t]{2,}", r", ", inLines[i].rstrip())
            raw[i] = inLines[i].split(", ")

            parsed = list(raw.values())

            f = open("decrypted.txt", "w", newline="\r\n")

            results = [int(i) for i in parsed[0]]

            for i in range(0, len(results)):
                ciphertext = decrypt(int(d), int(n), results[i])
                f.write(ciphertext)

            f.close()

assignment()

```