LAB EXERCISE

Week 04 exercise

Môn: Nhập môn mã hóa mật mã
Lớp: 19MMT
Tên: Nguyễn Thanh Quân
MSSV: 19127525

THỰC HÀNH

1. Ngôn ngữ: Python

2. Hướng dẫn chạy chương trình :

- Giả sử môi trường làm việc là Linux và folder bài làm được lưu trong thư mục ~/Desktop

Bước 1: Mở terminal

```
Ctrl + Shift + T
```

Bước 2: Giải nén file 19127525.zip

```
cd ~/Desktop && unzip -u ./19127525
```

Bước 3: Di chuyển vào folder 19127525

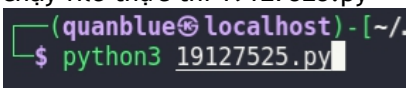
```
cd 19127525
```

Bước 4: Chạy file bài làm

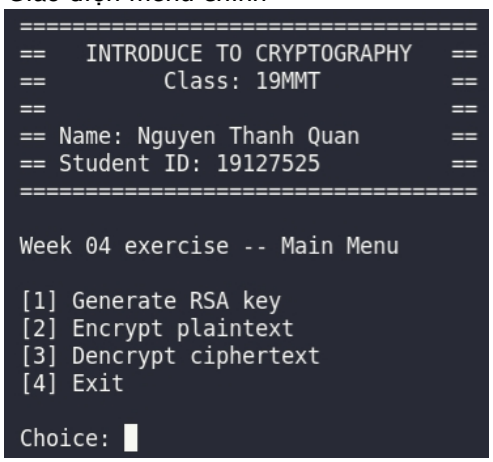
```
python3 19127525.py
```

3. Hướng dẫn sử dụng:

chạy file thực thi 19127525.py



Giao diện menu chính

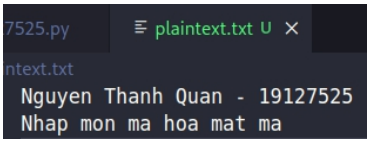


Chọn ‘1’ để tạo RSA key, chương trình sẽ yêu cầu nhập folder lưu trữ file, nếu folder đã có sẵn, chương trình sẽ ghi key vào trong folder, nếu không chương trình sẽ tạo folder mới rồi ghi key vào. Ở đây cho thư mục lưu vào folder test (chưa tạo)

```
==================================
==   INTRODUCE TO CRYPTOGRAPHY   ==
==         Class: 19MMT          ==
==                               ==
== Name: Nguyen Thanh Quan       ==
== Student ID: 19127525          ==
==================================

Week 04 exercise -- Main Menu

[1] Generate RSA key
[2] Encrypt plaintext
[3] Dencrypt ciphertext
[4] Exit

Choice: 1
----------------------------------
Input DIRECTORY: test

 > Generate key done!! <
Press any key to continue
```

Đây là thư mục ‘test’ chương trình đã tạo tự động và ghi vào 2 file chứa khóa công khai và bí mật

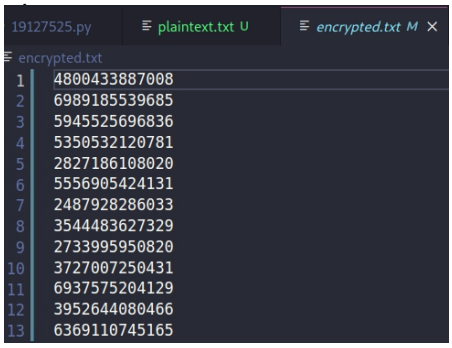
Đây là nội dung file plaintext.txt (file ví dụ) để chạy demo



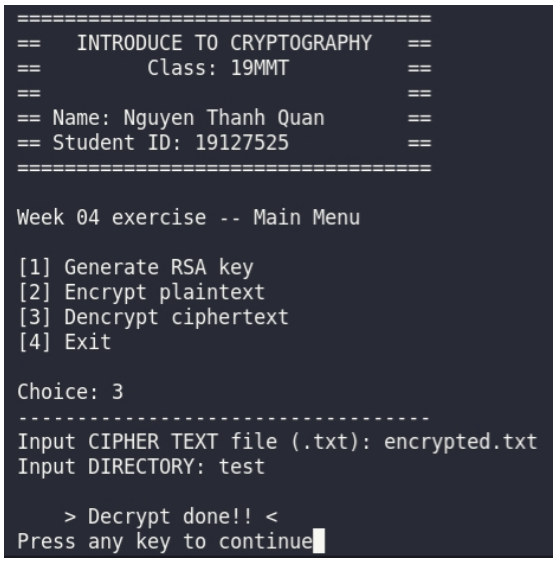
Chọn ‘2’ để mã hóa file plaintext. Chương trình sẽ yêu cầu nhập vào tên file plaintext cần mã hóa và thư mục chứa khóa. Ở đây là file file plaintext là plaintext.txt và thư mục chứa khóa là test (đã được tạo ở trên)

```
==================================
==   INTRODUCE TO CRYPTOGRAPHY   ==
==         Class: 19MMT          ==
==                               ==
== Name: Nguyen Thanh Quan       ==
== Student ID: 19127525          ==
==================================

Week 04 exercise -- Main Menu

[1] Generate RSA key
[2] Encrypt plaintext
[3] Dencrypt ciphertext
[4] Exit

Choice: 2
----------------------------------
Input PLAIN TEXT file (.txt): plaintext.txt
Input DIRECTORY: test

   > Encrypt done!! <
Press any key to continue
```

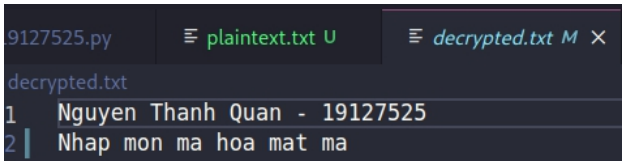
Sau khi nhập vào input, chương trình sẽ tạo file encryoted.txt và ghi nội dung đã được mã hóa của file plaintext.txt. Đây là nội dung của file encrypted.txt



Chọn '3' để giải mã (decrypt) file mã hóa, chương trình sẽ yêu cầu nhập vào file mã hóa (ciphertext) và thư mục chứa key. Ở đây file mã hóa là file encrypted.txt đã được mã hóa ở trên và thư mục chứa khóa là test



Sau khi nhập vào input, chương trình sẽ tạo file decryoted.txt và ghi nội dung đã được giải mã của file encrypted.txt. Đây là nội dung của file decrypted.txt



4. Cấu trúc file:

19127525/
|__Source/
| |__19127525.py
| |__plaintext01.txt
| |__plaintext02.txt
| |__encrypted.txt
| |__decrypted.txt
| |__QuansKey/
| | |__rsa_pub.txt
| | |__rsa.txt
|__Report/
   |__19127525.pdf

File có cấu trúc như sau cây thư mục ở trên
Trong đó báo cáo là file **19127525.pdf** nằm ở **./19127525/Report/19127525.pdf**
Các file mã nguồn nằm trong thư mục **Source** nằm ở **./19127525/Source** gồm có:

- file thực thi **19127525.py**
- các file muốn encrypt/decrypt (file chứa plaintext, thực hiện nhiệm vụ 2 - 3) là file **plaintext01.txt** và **plaintext02.txt** (2 file plaintext ví dụ) nằm cùng cấp với file thực thi
- 2 file **encrypted.txt** và **decrypted.txt** là 2 file mã hóa và giải mã plaintext ở trên, nằm cùng cấp với file thực thi
- Các cặp khóa sẽ được lưu trong 2 file **rsa.txt** và **rsa_pub.txt** vào thư mục con chứa khóa, ở đây là QuansKey (folder này là ví dụ). Các folder chứa khóa này nằm cùng cấp với file thực thi

5. Giải thích source code:

3 hàm chính **generate_key(directory)**, **encryption(plaintext, n, e)**, **decryption(plaintext, n, d)**

Hàm generate_key(directory):
- Input: *directory* (địa chỉ folder người dùng muốn lưu khóa)
- Output: cặp khóa sinh ra ghi vào các file *.txt* trong *directory*
- Thuật toán:
+ Bước 1: khởi tạo 2 số nguyên tố *p, q* sao cho $p \neq q$
+ Bước 2: tính *n, piN, e, d*
+ Bước 3: ghi các khóa vừa tìm được vào file *rsa_pub.txt* và *rsa.txt*

- Mã giả:

```
def generate_key(directory):
    # generate p, q
    q = generate_prime()

    do:
        p = generate_prime()
    while p == q

    # calulate n, piN, e
    n = p*q
    piN = (p-1)*(q-1)
    e = co_prime(piN)

    k = 1
    do:
    d = (k * piN + 1) div e
        k += 1
    while MulMod(d, e, piN) != 1

    # save key
    open(directory + '/rsa_pub.txt') as file:
        file.write(n, e)

    open(directory + '/rsa.txt') as file:
        file.write(n, d)
```

Hàm encryption(plaintext, n, e):
- Input: *list plaintext* lấy từ file có định dạng *.txt* do người dùng nhập vào, mỗi element trong list là 1 chuỗi thông điệp trên 1 dòng, *n* và *e* là khóa công khai được lấy từ file *rsa_pub.txt* trong folder người dùng chọn
- Output: file *encrypted.txt* (chứa thông điệp đã được mã hóa (ciphertext) từ các dữ liệu input)

- Thuật toán:

+ Bước 1: đọc từng kí tự trong *plaintext*, chuyển chúng thành *mã ascii*, mã này được định dạng kiểu str có độ dài là 3 (ví dụ: 'D' -> '068', 'd' -> '100').

+ Bước 2: Vì thông điệp (m) < n nên em sẽ cắt các thông điệp thành nhiều mảnh nhỏ, sao cho thông điệp đó có giá trị < n

. ví dụ: 'hello' -> '104101108108111'

n = 103025

Thông điệp sẽ chia thành ['104','101108','108','111']

+ Bước 3: ứng với mỗi thông điệp đã được chia nhỏ, áp dụng công thức $c = m^e \ mod \ n$ để mã hóa thông điệp đó

+ Bước 4: mở file *encrypted.txt* và ghi list dữ liệu đã mã hóa vào, mỗi thông điệp mã hóa viết trên 1 dòng

- Mã giả:

```
def encryption(plaintext, n, e):
    # convert plaintext into ascii
    ascii_text = []
    for message_text in plaintext:
        for i in message_text:
            ascii_num = str(ord(i))
              ascii_num = '0' * (3 - len(ascii_num)) + ascii_num
            ascii_text.append(ascii_num)

        ascii_text.append('010') # enter / newline

    # split message into smaller msg (int(msg) < n)
    ciphertext = []
    message = []
    i = 0
    while i < len(ascii_text):
        msg = ascii_text[i]
        while int(msg) < n:
             msg += ascii_text[i++]

        message.append(int(msg))

    # --------- encrypt ---------
    # c = m^e mod n
    for msg in message:
        c = PowerMod(msg,e,n)
        ciphertext.append(str(c) + '\n')

    # write encrypt message into file
    open('encrypted.txt') as file:
        file.writelines(ciphertext)
```

Hàm decryption(ciphertext, n, d):

- Input: *list ciphertext* lấy từ file có định dạng *.txt* do người dùng nhập vào, mỗi element trong list là 1 chuỗi thông điệp trên 1 dòng, *n* và *d* là khóa bí mật được lấy từ file *rsa.txt* trong folder người dùng chọn

- Output: file *decrypted.txt* (chứa thông điệp đã được giải mã (plaintext) từ các dữ liệu input)

- Thuật toán:

+ Bước 1: giải mã từng *mảnh ciphertext* trong *ciphertext* (mỗi mảnh được viết trên 1 dòng, theo định dạng của output của hàm encryption) theo công thức $m = c^d \ mod \ n$ và ghi vào *list message*, từ đó ta sẽ có list thông điệp được viết dưới dạng *mã ascii*

+ Bước 2: chuẩn hóa các mảnh thông điệp trong *list message* sao cho độ dài mỗi mảnh chia hết cho 3

. ví dụ: message = [‘72101’, ‘108108’, ’111’]
Standardized message = [‘072101’, ‘108108’, ’111’]

+ Bước 3: decode các *mảnh ascii message* đó sang *utf-8*, cứ 3 chữ số là 1 kí tự và ghép lại

. ví dụ: Standardized message = [‘072101’, ‘108108’, ’111’]
Decode: ‘072’-> ’H’, ‘101’ -> ‘e’, ‘108’ -> ‘l’, ‘111’ -> ‘o’
-> Plaintext: ‘Hello’

- Mã giả

```
def decryption(ciphertext, n, d):
    #----- decrypt -----
    message = []
    for c in ciphertext:
        # m = c^d mod n
        m = PowerMod(c,d,n)
        message.append(m)

    plaintext = ''

    for msg in message:
        # standardized msg with len(msg) % 3 == 0
        msg = '0'*add_zero + msg
        start = 0
        end = 2

        # decode ascii to utf-8 char
        while end < len(msg):
            c = chr(msg[start:end + 1])
            plaintext += c
            start += 3
            end += 3

    # write decrypt message into file
    open('decrypted.txt') as file:
        file.write(plaintext)
```