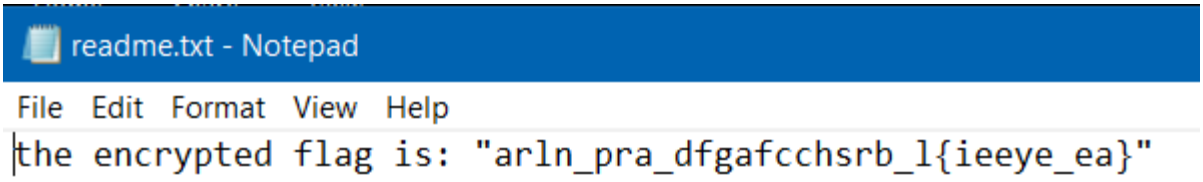
Fence

Chạy thử chương trình

```
┌──(kali㉿ DESKTOP-6TCCJEL)-[/mnt/c/Users/BILL/Desktop/Release_3/fence]
└─$ ./encryptor fasdfs
ssfdaf
```

Đọc file readme.txt



Như vậy ta cần nhập vào một chuỗi mà sau khi encrypt chương trình xuất ra

"arln_pra_dfgafcchsrb_l{ieeye_ea}"

Phân tích với ida, ta thấy đoạn xuất output ở phía gần cuối chương trình

```
60  std::operator+<char>(v20, v19, v17);
61  std::operator+<char>(v21, v20, v18);
62  v10 = std::operator<<<char>(&std::cout, v21);
63  std::ostream::operator<<(v10, &std::endl<char,std::char_traits<char>>);
```

Chương trình xuất ra chuỗi v21 = v20 + v18 = v19 + v17 + v18

ĐẦu tiên chương trình sẽ lấy chuỗi từ argv[1] và gán vào v16

```
std::allocator<char>::allocator(&v12, argv, envp);
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(v16, argv[1], &v12);
```

Đoạn code phía dưới để tạo ra chuỗi v19, v17, v18

```
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(v17);
for ( i = 0LL; ; i += 3LL )
{
  v4 = std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::length(v16);
  if ( i >= v4 )
    break;
  v5 = (char *)std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::operator[](v16, i);
  std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::operator+=(v17, (unsigned int)*v5);
}
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(v18);
for ( j = 1LL; ; j += 3LL )
{
  v6 = std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::length(v16);
  if ( j >= v6 )
    break;
  v7 = (char *)std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::operator[](v16, j);
  std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::operator+=(v18, (unsigned int)*v7);
}
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(v19);
for ( k = 2LL; ; k += 3LL )
{
  v8 = std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::length(v16);
  if ( k >= v8 )
    break;
  v9 = (char *)std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::operator[](v16, k);
  std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::operator+=(v19, (unsigned int)*v9);
}
```

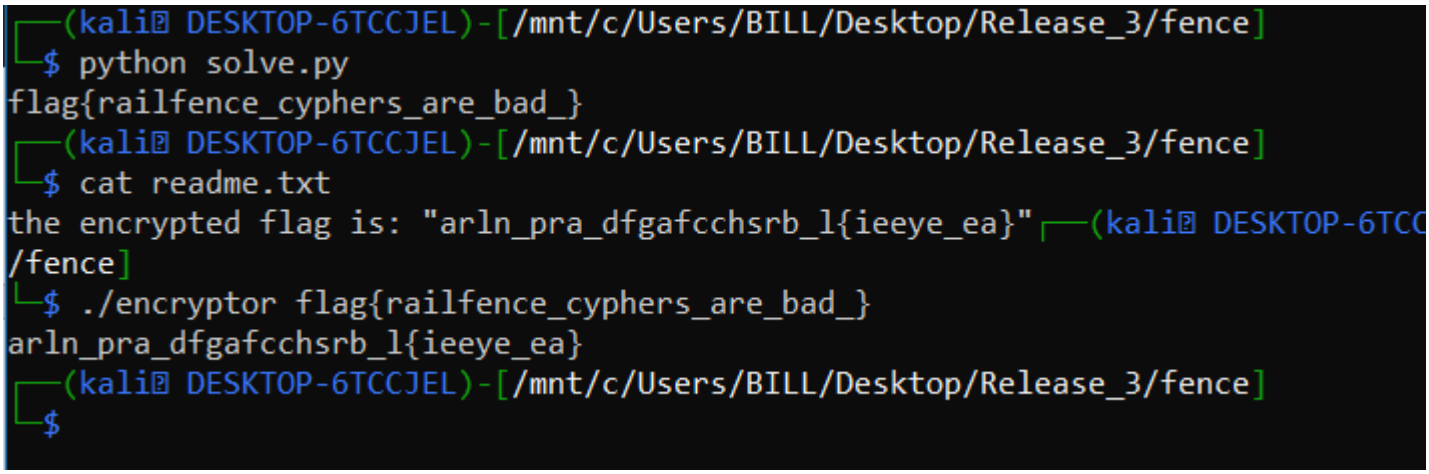
3 đoạn code khá giống nhau, chỉ khác ở vị trí bắt đầu (0, 1, 2), sau đó lặp đến hết chuỗi v16, mỗi lần lặp cách nhau 3, sau đó cộng thêm giá trị của chuỗi v16 vào phía cuối các chuỗi v17, v18, v19. Sau đó v21 = v19 + v17 + v18.

Từ đó ta có thể suy ngược ra flag từ: "arln_pra_dfgafcchsrb_l{ieeye_ea}" như sau:

solve.py

```python
encrypted = 'arln_pra_dfgafcchsrb_l{ieeye_ea}'

v19 = encrypted[:10]
v17 = encrypted[10:21]
v18 = encrypted[21:]

flag = [0] * 32

for i in range(len(v17)):
    flag[i * 3] = v17[i]

for i in range(len(v18)):
    flag[i * 3 + 1] = v18[i]

for i in range(len(v19)):
    flag[i * 3 + 2] = v19[i]

print(''.join(flag))

```

Độ dài của encrypted là 32, vì vậy v17, v18 có độ dài là 11 (vì bắt đầu từ 0 và 1), v19 có độ dài là 10 (vì bắt đầu từ 2)



Flag: flag{railfence_cyphers_are_bad_}