**TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHỆ**

**KHOA CÔNG NGHỆ THÔNG TIN**

**Môn: Bảo Mật Thông Tin**

Bài thực hành số 2



**Bài 1: Viết chương trình mã hóa và giãi mã văn bản với thuật toán mã hóa DES. Chương trình có thể thực hiện các chức năng sau:**

**Cho phép nhập văn bản vào hệ thống. Cho phép nhập khóa bảo vệ văn bản.**

**Cho phép ghi File và mở File.**

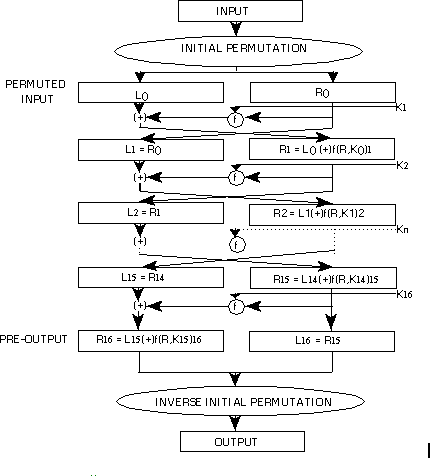
# Hướng dẫn mã hóa DES:

DES Là một hệ mật mã được sử dụng rộng rãi nhất trên thế giới. DES được IBM phát triển vào những năm 1970 và được xem như cải biên của hệ mật mã LUCIPHER. DES được chấp nhận bởi National Bureau of Standards, ngày nay gọi là NIST (National Institute of Standards and Technology). DES trở thành chuẩn mã hóa dữ liệu chính thức của chính phủ Hoa Kỳ vào năm 1977.

*Mô tả thuật toán:*

DES là thuật toán mã hóa khối (block cipher), mỗi khối dữ liệu có độ dài 64 bit. Một block bản gốc sau khi mã hóa tạo ra một block bản mã. Quá trình mã hóa và giải mã đều dùng chung một khóa.

Khóa có độ dài là 56 bit, cộng thêm 8 bit chẵn lẻ được sử dụng để kiểm soát lỗi. Các bit chẵn lẻ nằm ở các vị trí 8, 16, 24…64. tức là cứ 8 bit thì có một bit kiểm soát lỗi .

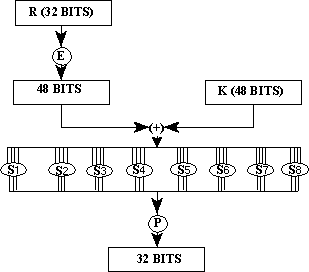


Hình 1: Sơ đồ hoạt động của DES

Theo sơ đồ họat động của DES như trên ta có thể thấy:

DES thực hiện trên từng block bản gốc. Sau khi thực hiện hoán vị khởi đầu (Initial Permutation – IP) khối dữ liệu được chia làm hai nửa trái và phải, mỗi nửa 32 bit. Quá trình được lặp lại qua 16 vòng, mỗi vòng là một hàm f. Sau 16 vòng lặp, hai nửa trái và phải được kết hợp lại và thực hiện hoán vị cuối cùng (hoán vị ngược – Inverse Initial Permutation) để kết thúc thuật toán.

Mỗi vòng của DES được thực hiện theo các bước sau:



Hình 2: Một vòng hoạt động của DES

*Bước 1:* Sử dụng hoán vị khởi đầu để thay đổi thứ tự các bit.

**Bảng P3.B10: Bảng hoán vị khởi đầu:** (hoán vị bit 1 thành bit 58, bit 2 thành bit 50….)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 58 | 50 | 42 | 34 | 26 | 18 | 10 | 2 | 60 | 52 | 44 | 36 | 28 | 20 | 12 | 4 |
| 62 | 54 | 46 | 38 | 30 | 22 | 14 | 6 | 64 | 56 | 48 | 40 | 32 | 24 | 16 | 8 |
| 57 | 49 | 41 | 33 | 25 | 17 | 9 | 1 | 59 | 51 | 43 | 35 | 27 | 19 | 11 | 3 |
| 61 | 53 | 45 | 37 | 29 | 21 | 13 | 5 | 63 | 55 | 47 | 39 | 31 | 23 | 15 | 7 |

*Bước 2*: Bản gốc được chia làm hai nửa trái và phải, mỗi nửa 32 bit.

*Bước 3:* Ban đầu khóa 64 bit được bỏ đi 8 bit kiểm soát lỗi. Sự lọai bỏ được thực hiện theo bảng sau:

# Bảng 1:Bảng loại bỏ 8 bit kiểm soát lỗi

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 57 | 49 | 41 | 33 | 25 | 17 | 9 | 1 | 58 | 50 | 42 | 34 | 26 | 18 |
| 10 | 2 | 59 | 51 | 43 | 35 | 27 | 19 | 11 | 3 | 60 | 52 | 44 | 36 |
| 63 | 55 | 47 | 39 | 31 | 23 | 15 | 7 | 62 | 54 | 46 | 38 | 30 | 22 |
| 14 | 6 | 61 | 53 | 45 | 37 | 29 | 21 | 13 | 5 | 28 | 20 | 12 | 4 |

Sau đó khóa được chia làm hai nửa, mỗi nửa 28 bit.

*Bước 4:* Các nửa của khóa lần lượt được dịch trái (số bit dịch là 1 hay 2 tùy theo vòng thực hiện).

# Bảng 2: Bảng dịch :

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Vòng | 1 2 3 4 5 6 7 8 9 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| Số bit dịch | 1 1 2 2 2 2 2 2 1 2 | 2 | 2 | 2 | 2 | 2 | 1 |

Các nửa được ghép lại với nhau, hoán vị và chọn ra 48 bit bằng cách đổi chỗ các bit theo bảng hoán vị nén - compression permutation ( hay còn gọi là hoán vị lựa chọn- permuted choice):

**Bảng 3: Bảng hoán vị nén**: (bit ở vị trí 14 của khóa dịch được chuyển tới vị trí 1 của đầu ra, bit ở vị trí 17 của khóa dịch được chuyển tới vị trí 2 của đầu ra,…, bit thứ 18 bị lọai bỏ…)

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 14 | 17 | 11 | 24 | 1 | 5 | 3 | 28 | 15 | 6 | 21 | 10 |
| 23 | 19 | 12 | 4 | 26 | 8 | 16 | 7 | 27 | 20 | 13 | 2 |
| 41 | 52 | 31 | 37 | 47 | 55 | 30 | 40 | 51 | 45 | 33 | 48 |
| 44 | 49 | 39 | 56 | 34 | 53 | 46 | 42 | 50 | 36 | 29 | 32 |

*Bước 5:* 32 bit của bản gốc bên phải được mở rộng thành 48 bit để XOR với 48 bit khóa. Khối bit này lại thực hiện hoán vị một lần nữa, thay đổi thứ tự các bit bằng cách lặp lại một số bit ( hoán vị mở rộng - Expansion Permutation).

**Bảng 4: Bảng hoán vị mở rộng – hộp E** ( bit ở vị trí thứ 32 của khối dữ liệu vào được chuyển tới vị trí thứ nhất trong khối dữ liệu ra, bit ở vị trí thứ 4 của khối dữ liệu vào được chuyển tới vị trí thứ 5 và 7 trong khối dữ liệu ra,…)

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 32 | 1 | 2 | 3 | 4 | 5 | 4 | 5 | 6 | 7 | 8 | 9 |
| 8 | 9 | 10 | 11 | 12 | 12 | 12 | 13 | 14 | 15 | 16 | 17 |
| 16 | 17 | 18 | 19 | 20 | 21 | 20 | 21 | 22 | 23 | 24 | 25 |
| 24 | 25 | 26 | 27 | 28 | 29 | 28 | 29 | 30 | 31 | 32 | 1 |

*Bước 6:* kết quả của bước 3 và bước 5 được XOR với nhau.

*Buớc 7*: Kết quả của bước 6 được chuyển thành 32 bit bằng cách sử dụng hàm thay thế và lựa chọn.

Sự thay thế được thực hiện bởi 8 hộp thay thế (substitution boxes, S-boxes). Khối 48 bit được chia thành 8 khối 6 bit. Mỗi khối được thực hiện trên một hộp S riêng biệt (separate S- box): khối 1 được thực hiện trên hộp S1, khối 2 được thực hiện trên hộp S2…

Mỗi hộp S là một bảng gồm 4 hàng và 16 cột. Mỗi phần tử của hộp là một số 4 bit. Với sáu bit vào hộp S sẽ xác định được số hàng và số cột để tìm ra kết quả.

Cách thức xác định kết quả: nhận vào 6 bit lần lượt là b1, b2, b3, b4, b5, và b6. Bit b1và b6 được kết hợp lại thành một số 2 bit tương ứng với số hàng trong bảng (có giá trị từ 0 đến 3). Bốn bit ở giữa được kết hợp lại thành một số 4 bit tương ứng với số cột trong bảng ( nhận giá trị từ 0 đến 15).

Ví dụ : Dùng hộp S thứ 6. Nếu dữ liệu nhận vào là 110010. Bit đầu tiên kết hợp với bit cuối tạo thành 10 (khi đổi sang số thập phân có giá trị bằng 2 tương ứng với hàng thứ 2). Bốn bit giữa kết hợp lại thành 1001(khi đổi sang số thập phân có giá trị bằng 9 tương ứng với cột thứ 9) => Giá trị cần tìm hàng 2 cột 9 là 0. Như vậy giá trị 0000 được thay thế cho 110010.

Dùng hộp S thứ nhất. Nếu dữ liệu nhận vào là 011011. Bit đầu tiên kết hợp với bit cuối tạo thành 01 (khi đổi sang số thập phân có giá trị bằng 1 tương ứng với hàng 1). Bốn bit giữa kết hợp lại thành 1101(khi đổi sang số thập phân có giá trị bằng 13 tương ứng với cột thứ 13) => Giá trị cần tìm hàng 1 cột 13 là 5. Như vậy giá trị 0101 được thay thế cho 011011.

# Bảng 5: Bảng hộp S:

Hộp S thứ nhất.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 14 | 4 | 13 | 1 | 2 | 15 | 11 | 8 | 3 | 10 | 6 | 12 | 5 | 9 | 0 | 7 |
| 0 | 15 | 7 | 4 | 14 | 2 | 13 | 1 | 10 | 6 | 12 | 11 | 9 | 5 | 3 | 8 |
| 4 | 1 | 14 | 8 | 13 | 6 | 2 | 11 | 15 | 12 | 9 | 7 | 3 | 10 | 5 | 0 |
| 15 | 12 | 8 | 2 | 4 | 9 | 1 | 7 | 5 | 11 | 3 | 14 | 10 | 0 | 6 | 13 |
| Hộp S thứ hai. | | | | | | | | | | | | | | | |
| 15 | 1 | 8 | 14 | 6 | 11 | 3 | 4 | 9 | 7 | 2 | 13 | 12 | 0 | 5 | 10 |
| 3 | 13 | 4 | 7 | 15 | 2 | 8 | 14 | 12 | 0 | 1 | 10 | 6 | 9 | 11 | 5 |
| 0 | 14 | 7 | 11 | 10 | 4 | 13 | 1 | 5 | 8 | 12 | 6 | 9 | 3 | 2 | 15 |
| 13 | 8 | 10 | 1 | 3 | 15 | 4 | 2 | 11 | 6 | 7 | 12 | 0 | 5 | 14 | 9 |
| Hộp S thứ ba. | | | | | | | | | | | | | | | |
| 10 | 0 | 9 | 14 | 6 | 3 | 15 | 5 | 1 | 13 | 12 | 7 | 11 | 4 | 2 | 8 |
| 13 | 7 | 0 | 9 | 3 | 4 | 6 | 10 | 2 | 8 | 5 | 14 | 12 | 11 | 15 | 1 |
| 13 | 6 | 4 | 9 | 8 | 15 | 3 | 0 | 11 | 1 | 2 | 12 | 5 | 10 | 14 | 7 |
| 1 | 10 | 13 | 0 | 6 | 9 | 8 | 7 | 4 | 15 | 14 | 3 | 11 | 5 | 2 | 12 |
| Hộp S thứ tư. | | | | | | | | | | | | | | | |
| 7 | 13 | 14 | 3 | 0 | 6 | 9 | 10 | 1 | 2 | 8 | 5 | 11 | 12 | 4 | 15 |
| 13 | 8 | 11 | 5 | 6 | 15 | 0 | 3 | 4 | 7 | 2 | 12 | 1 | 10 | 14 | 9 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 6 | 9 | 0 | 12 | 11 | 7 | 13 | 15 | 1 | 3 | 14 | 5 | 2 | 8 | 4 |
| 3 | 15 | 0 | 6 | 10 | 1 | 13 | 8 | 9 | 4 | 5 | 11 | 12 | 7 | 2 | 14 |
| Hộp S thứ năm. | | | | | | | | | | | | | | | |
| 2 | 12 | 4 | 1 | 7 | 10 | 11 | 6 | 8 | 5 | 3 | 15 | 13 | 0 | 14 | 9 |
| 14 | 11 | 2 | 12 | 4 | 7 | 13 | 1 | 5 | 0 | 15 | 10 | 3 | 9 | 8 | 6 |
| 4 | 2 | 1 | 11 | 10 | 13 | 7 | 8 | 15 | 9 | 12 | 5 | 6 | 3 | 0 | 14 |
| 11 | 8 | 12 | 7 | 1 | 14 | 2 | 13 | 6 | 15 | 0 | 9 | 10 | 4 | 5 | 3 |

Hộp S thứ sáu.

12 1 10 15 9 2 6 8 0 13 3 4 14 7 5 11

10 15 4 2 7 12 9 5 6 1 13 14 0 11 3 8

9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6

4 3 2 12 9 5 15 10 11 14 1 7 6 0 8 13

Hộp S thứ bảy.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4 | 11 | 2 | 14 | 15 | 0 | 8 | 13 | 3 | 12 | 9 | 7 | 5 | 10 | 6 | 1 |
| 13 | 0 | 11 | 7 | 4 | 9 | 1 | 10 | 14 | 3 | 5 | 12 | 2 | 15 | 8 | 6 |
| 1 | 4 | 11 | 13 | 12 | 3 | 7 | 14 | 10 | 15 | 6 | 8 | 0 | 5 | 9 | 2 |
| 6 | 11 | 13 | 8 | 1 | 4 | 10 | 7 | 9 | 5 | 0 | 15 | 14 | 2 | 3 | 12 |
| Hộp S thứ tám. | | | | | | | | | | | | | | | |
| 13 | 2 | 8 | 4 | 6 | 15 | 11 | 1 | 10 | 9 | 3 | 14 | 5 | 0 | 12 | 7 |
| 1 | 15 | 13 | 8 | 10 | 3 | 7 | 4 | 12 | 5 | 6 | 11 | 0 | 14 | 9 | 2 |
| 7 | 11 | 4 | 1 | 9 | 12 | 14 | 2 | 0 | 6 | 10 | 13 | 15 | 3 | 5 | 8 |
| 2 | 1 | 14 | 7 | 4 | 10 | 8 | 13 | 15 | 12 | 9 | 0 | 3 | 5 | 6 | 11 |

Kết quả của sự thay thế là 8 khối 4 bit được sinh ra, chúng được kết hợp lại thành một khối 32 bit. Khối này được chuyển tới bước tiếp theo: hộp hoán vị P (P-box permutation). Hoán vị ở bước này ánh xạ mỗi bit dữ liệu vào tới một vị trí trong khối dữ liệu ra, không có bit nào bị bỏ qua cũng như được sử dụng hai lần. nó còn được gọi là hoán vị trực tiếp (straight permutation).

**Bảng 6: Bảng hộp hoán vị P** cho biết vị trí của mỗi bit cần chuyển (bit 1 chuyển tới bit 16, bit 2 chuyển tới bit 7…)

16 7 20 21 29 12 28 17 1 15 23 26 5 18 31 10

2 8 24 14 32 27 3 9 19 13 30 6 22 11 4 25

*Buớc 8*: Kết quả của bước 7 được XOR với nửa trái 32 bit được tạo ra ở bước 2.

*Bước 9:* kết quả tạo ra ở bước 8 trở thành nửa phải mới, nửa phải cũ (tạo ở bước 2) trở thành nửa trái mới.

Sau khi thực hiện hết 16 vòng lặp hoán vị cuối cùng được thực hiện để kết thúc thuật toán. Hoán vị cuối cùng là nghịch đảo của hoán vị khởi đầu.

# Bảng 7: Bảng hoán vị cuối

40 8 48 16 56 24 64 32 39 7 47 15 55 23 63 31

38 6 46 14 54 22 62 30 37 5 45 13 53 21 61 29

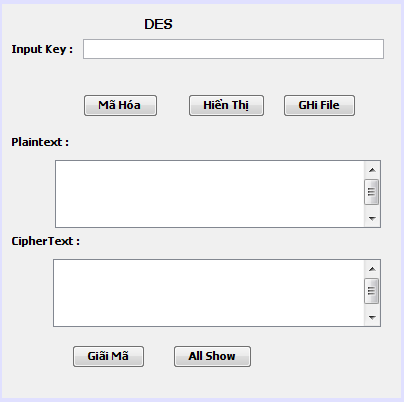
36 4 44 12 52 20 60 28 35 3 43 11 51 19 59 27

34 2 42 10 50 18 58 26 33 1 41 9 49 17 57 25

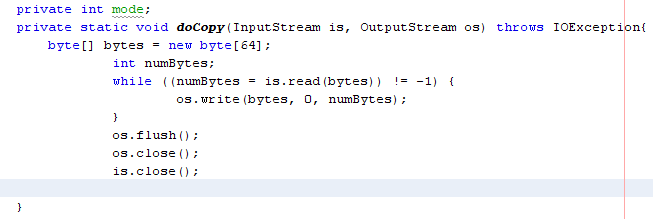
Các chế độ họat động của DES

Có bốn chế độ làm việc đã đuợc phát triển cho DES:

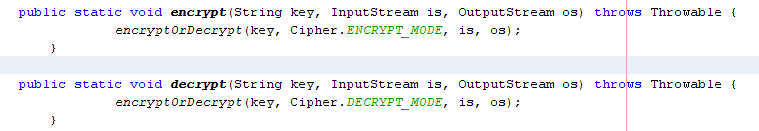
* Chế độ sách mã điện tử (ECB).
* Chế độ phản hồi mã (CFB).
* Chế độ liên kết khối mã (CBC – Cipher Block Chaining).
* Chế độ phản hồi đầu ra (OFB).
  1. **Hướng dẫn thực hành Bước 1**: Thiết Kế Form :



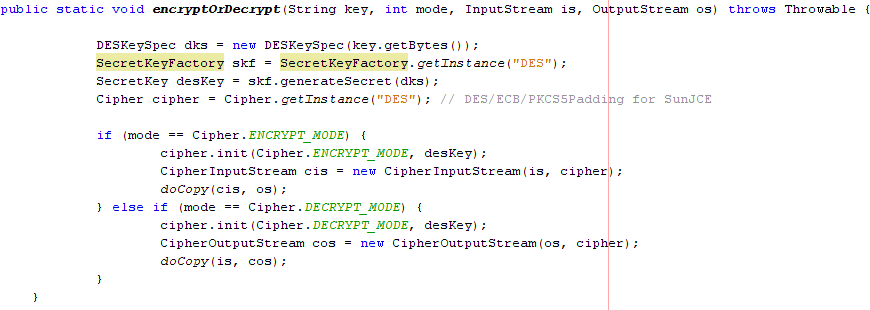
**Bước 2**: Viết hàm xử lý sự kiện B2.1 Hàm doCopy



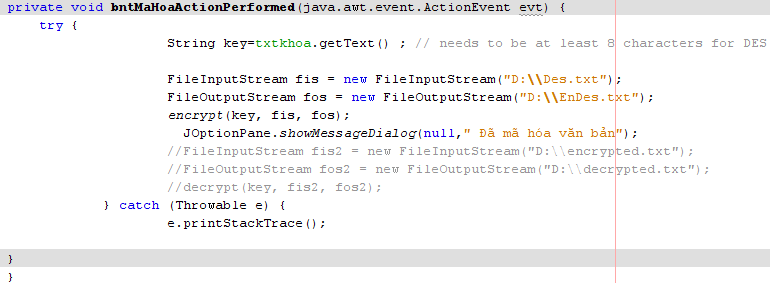
B2.2 Hàm mã Hóa và giải mã



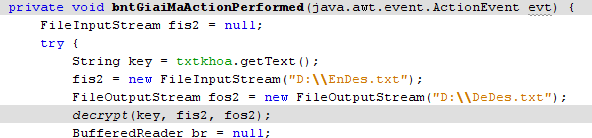
B2.3 Hàm thưc hiện

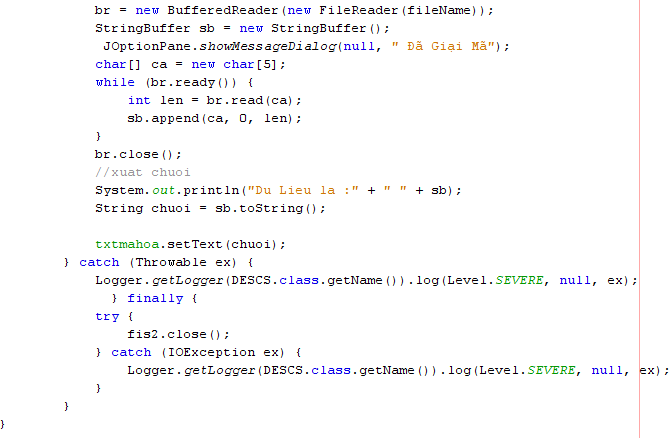


B2.4 Viết chức năng Mã Hóa



B2.5 Viết Chức năng Giãi Mã

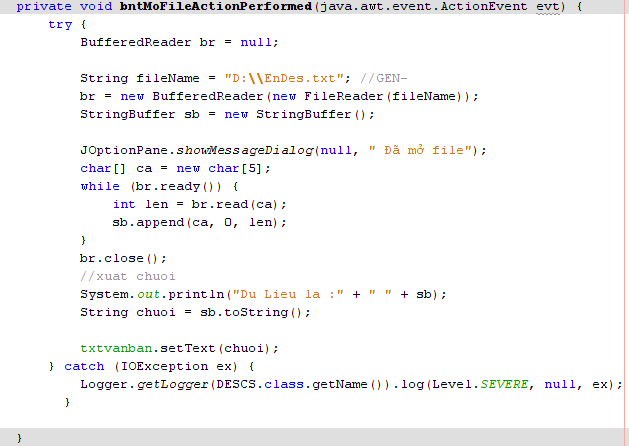




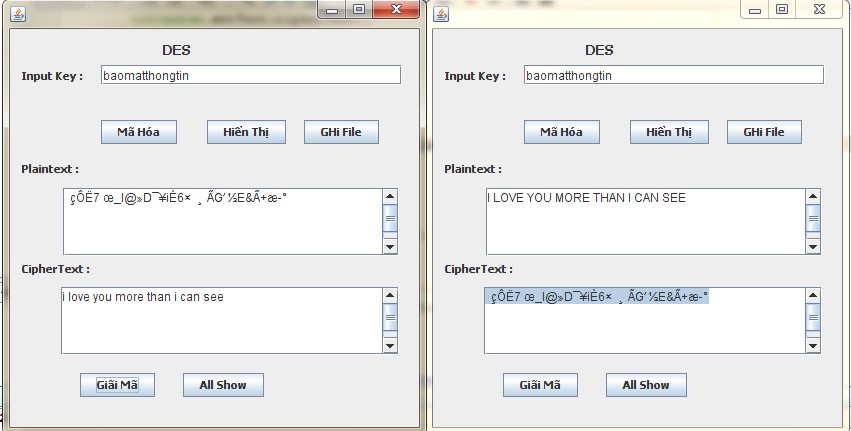
B2.6 Viết chức năng Ghi FILE



B2.7 Viết chức năng Mở FILE



# Bước 3: Kiểm Tra



**Bài 2: Viết chương trình mã hóa và giãi mã văn bản với thuật toán mã hóa 3DES. Chương trình có thể thực hiện các chức năng sau:**

**Cho phép nhập văn bản vào hệ thống. Cho phép nhập khóa bảo vệ văn bản.**

**Cho phép ghi File và mở File.**

* 1. **Hướng dẫn thuật toán TRIPLEDES:**

TripleDES một biến thể an toàn hơn của DES còn được gọi là DESede hay 3DES. TripleDES có tính bảo mật cao hơn DES do sử dụng 3 vòng DES với các khóa khác nhau. Vòng đầu tiên và vòng thứ ba là vòng mã hóa, vòng thứ hai là vòng giải mã. DESede có thể dùng hai hoặc ba khóa có độ dài 56, 112 hoặc 168. nếu dùng hai khóa thì khóa đầu tiên được dùng cho vòng thứ nhất và vòng thứ ba, khóa thứ hai dùng cho vòng thứ hai.

* Mã hóa với ba khóa 56 bit ( 168 bit).

# Bảng B1: 3DES Mã hóa với ba khóa 56 bit

|  |  |
| --- | --- |
| NGƯỜi GỬI | NGƯỜI NHẬN |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Bước**  nhất | **1**: | mã | hóa | plaintext | bằng | khóa | **Bước 1**:giải mã bản mã với khóa thứ ba |
| **Bước 2**: mã hóa văn bản được tạo ra ở  bước 1 bằng khóa thứ hai | | | | | | | **Bước 2**: giải mã văn bản được tạo ra ở  bước 1 bằng khóa thứ hai |
| **Bước 3**: mã hóa văn bản được tạo ra ở bước 2 bằng khóa thứ ba, tạo ra bản  mã gửi cho người nhận. | | | | | | | **Bước 3**: giải mã văn bản được tạo ra ở bước 2 bằng khóa thứ nhất, tạo ra  bản gốc do người gửi gửi. |

* Mã hóa với hai khóa 56 bit ( 112 bit)

# Bảng B2: 3DES Mã hóa với hai khóa 56 bit

|  |  |
| --- | --- |
| NGƯỜi GỬI | NGƯỜI NHẬN |
| **Bước 1**: mã hóa plaintext bằng khóa  nhất | **Bước 1**:giải mã bản mã với khóa thứ  nhất |
| **Bước 2**: giải mã văn bản được tạo ra ở  bước 1 bằng khóa thứ hai | **Bước 2**: mã hóa văn bản được tạo ra ở  bước 1 bằng khóa thứ hai |
| **Bước 3**: mã hóa văn bản được tạo ra ở b  2 bằng khóa thứ nhất, tạo ra bản mã gửi người nhận. | **Bước 3**: giải mã văn bản được tạo ra ở bước 2 bằng khóa thứ nhất, tạo ra  bản gốc do người gửi gửi. |

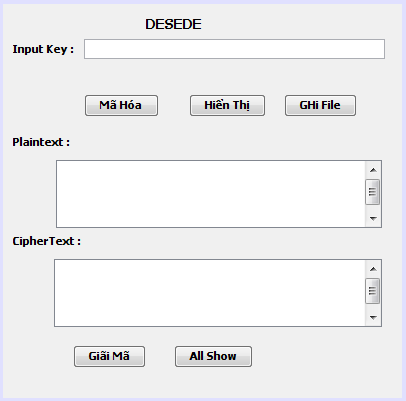
* Mã hóa với một khóa 56 bit

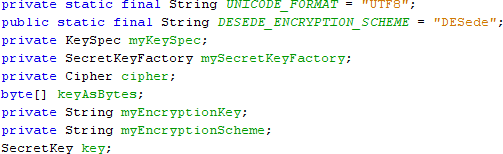
# Bảng B3: 3DES Mã hóa với một khóa 56 bit

|  |  |
| --- | --- |
| NGƯỜi GỬI | NGƯỜI NHẬN |
| **Bước 1**: mã hóa plaintext | **Bước 1**:giải mã bản mã nhận được từ người gửi. |
| **Bước 2:** giải mã văn bản được tạo ra ở bước 1 |  |
| **Bước 3**: mã hóa văn bản được tạo ra ở bước, tạo ra bản mã gửi cho người nhận. |  |

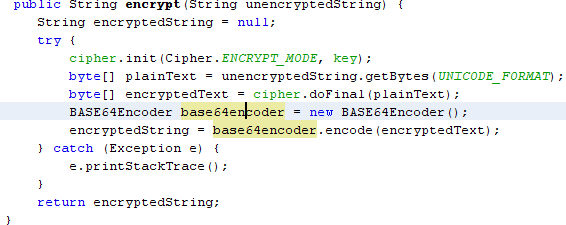
Mặc dù 3DES có tính bảo mật cao hơn DES, nhưng thực tế ít được sử dụng vì để tạo ra được bản mã phải chạy ba lần DES, nên tốc độ chậm, chiếm nhiều tài nguyên.

* 1. **Hướng dẫn thực hành Bước 1**: Thiết Kế Form :

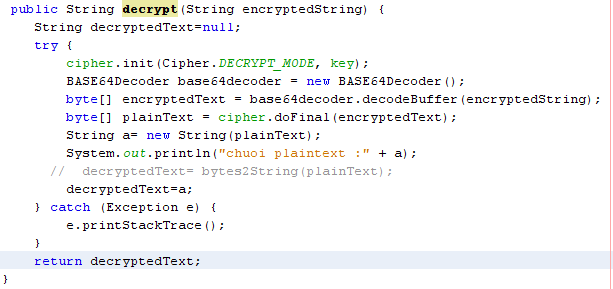


**Bước 2**: Viết hàm xử lý sự kiện B2.1: Khai báo các biến sau

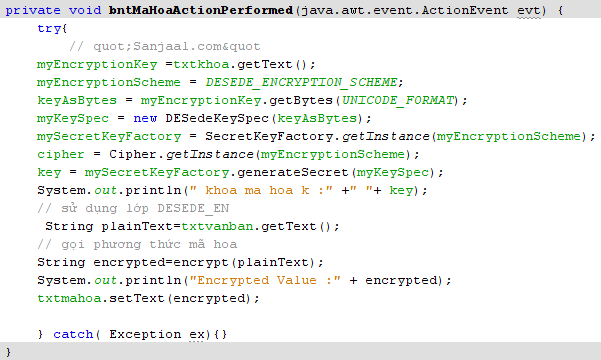
B2.2: Viết phương thức mã hóa encrypt



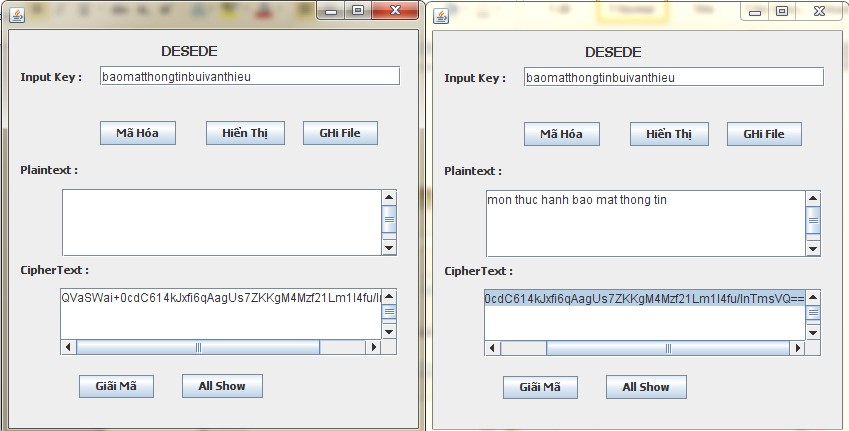
B2.3: Viết phương thức giãi mã decrypt

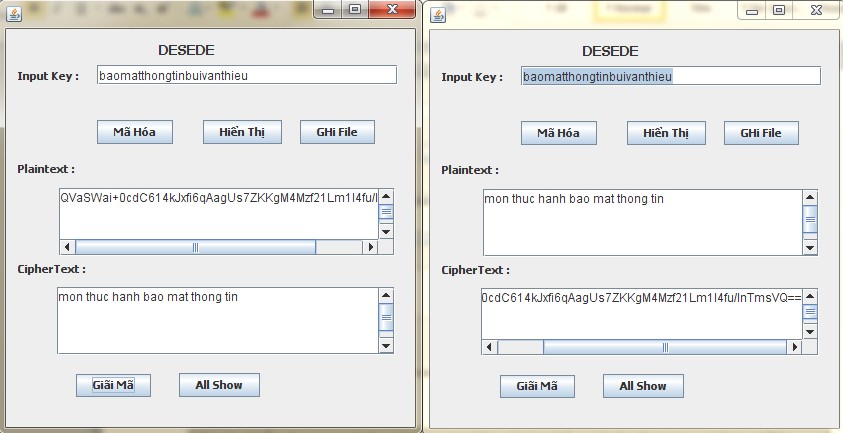


B2.4 Viết hàm xử lý sự kiện mã

hóa 

# Bước 3: Kiểm Tra





**Bài 3: Viết chương trình mã hóa và giãi mã văn bản với thuật toán mã hóa AES. Chương trình có thể thực hiện các chức năng sau:**

**Cho phép nhập văn bản vào hệ thống. Cho phép nhập khóa bảo vệ văn bản.**

**Cho phép ghi File và mở File.**

* 1. **Hướng dẫn thuật toán AES:**

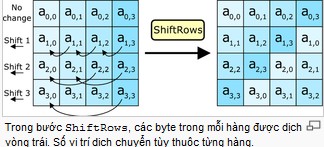
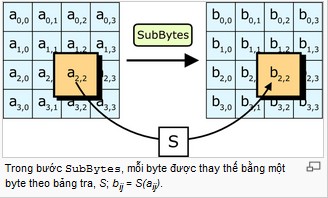
AES chỉ làm việc với khối dữ liệu 128 bít và khóa có độ dài 128, 192 hoặc 256 bít . Các khóa con sử dụng trong các chu trình được tạo ra bởi quá trình tạo khóa con Rijndael Rijndael có thể làm việc với dữ liệu và khóa có độ dài bất kỳ là bội số của 32 bít nằm trong khoảng từ 128 tới 256 bít.

Hầu hết các phép toán trong thuật toán AES đều thực hiện trong một trường hữu hạn.

AES làm việc với từng khối dữ liệu 4×4 byte (tiếng Anh: *state*, khối trong Rijndael có thể có thêm cột). Quá trình mã hóa bao gồm 4 bước:

* + 1. AddRoundKey — mỗi byte của khối được kết hợp với khóa con, các khóa con này được tạo ra từ quá trình tạo khóa con Rijndael.
    2. SubBytes — đây là phép thế (phi tuyến) trong đó mỗi byte sẽ được thế bằng một byte khác theo bảng tra (Rijndael S-box).
    3. ShiftRows — đổi chỗ, các hàng trong khối được dịch vòng.

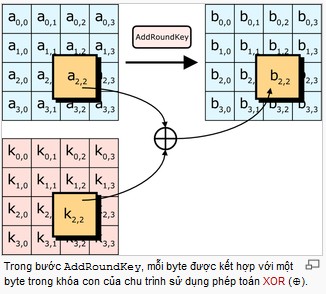
nghịch. Hộp S-box này cũng được chọn để tránh các điểm bất động (fixed point).



**Bước ShiftRows**

Các hàng được dịch vòng một số vị trí nhất định. Đối với AES, hàng đầu được giữ nguyên. Mỗi byte của hàng thứ 2 được dịch trái một vị trí. Tương tự, các hàng thứ 3 và 4 được dịch 2 và 3 vị trí. Do vậy, mỗi cột khối đầu ra của bước này sẽ bao gồm các byte ở đủ 4 cột khối đầu vào. Đối với Rijndael với độ dài khối khác nhau thì số vị trí dịch chuyển cũng khác nhau.

* + 1. MixColumns — quá trình trộn làm việc theo các cột trong khối theo một phép biến đổi tuyến tính.

Tại chu trình cuối thì bước MixColumns được thay thế bằng bướcAddRoundKey

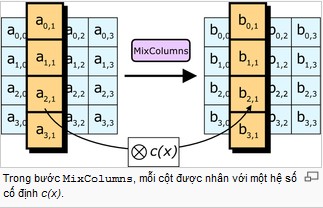
**Bước AddRoundKey:**

Tại bước này, khóa con được kết hợp với các khối. Khóa con trong mỗi chu trình được tạo ra từ khóa chính với quá trình tạo khóa con Rijndael; mỗi khóa con có độ dài giống như các khối. Quá trình kết hợp được thực hiện bằng cách XOR từng bít của khóa con với khối dữ liệu.

**Bước SubBytes**

Các byte được thế thông qua bảng tra S- box. Đây chính là quá trình phi tuyến của thuật toán. Hộp S-box này được tạo ra từ một phép nghịch đảotrong trường hữu hạn GF (*28*) có tính chất phi tuyến. Để chống lại các tấn công dựa trên các đặc tính đại số, hộp S-box này được tạo

nên bằng cách kết hợp phép nghịch đảo với một phép biến đổi affine khả

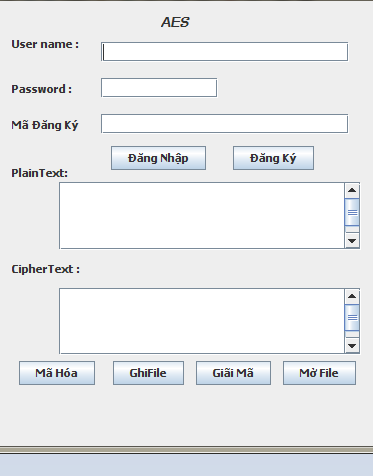


**Bước MixColumns**

Bốn byte trong từng cột được kết hợp lại theo một phép biến đổi tuyến tính khả nghịch. Mỗi khối 4 byte đầu vào sẽ cho một khối 4 byte ở đầu ra với tính chất là mỗi byte ở đầu vào đều ảnh hưởng tới cả 4 byte đầu ra. Cùng với bước ShiftRows, MixColumns đã tạo ra tính chất khuyếch tán cho thuật toán. Mỗi cột được xem như một đa thức trong trường hữu hạn và được nhân với đa thức *c*(*x*) = 3*x*3 + *x*2 + *x* + 2 (modulo *x*4 + 1). Vì thế, bước này có thể được xem là phép nhân ma trận trong trường hữu hạn.

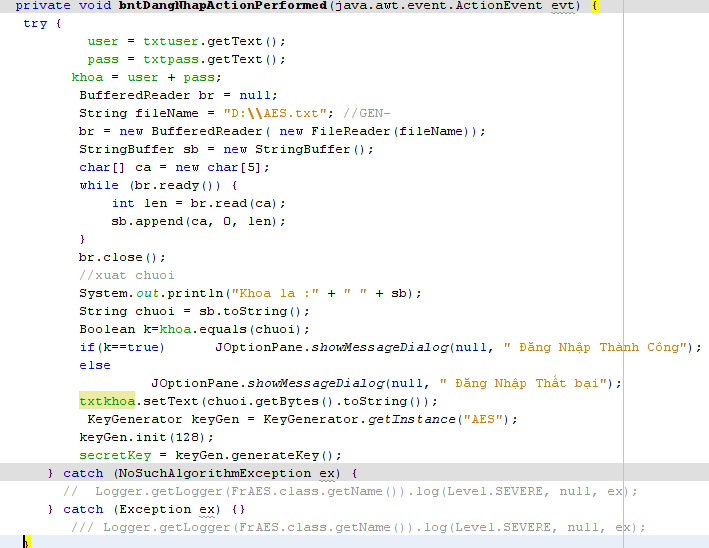
**Tóm lại:** Thuật toán AES có khối dữ liệu 128 bit , độ dài khóa 128,192, 256 bit , cấu trúc mạng thay thế-hoán vị , số chu trình 10,12,14 tùy theo độ dài khóa.

* 1. **Hướng dẫn thực hành Bước 1**: Thiết Kế Form :

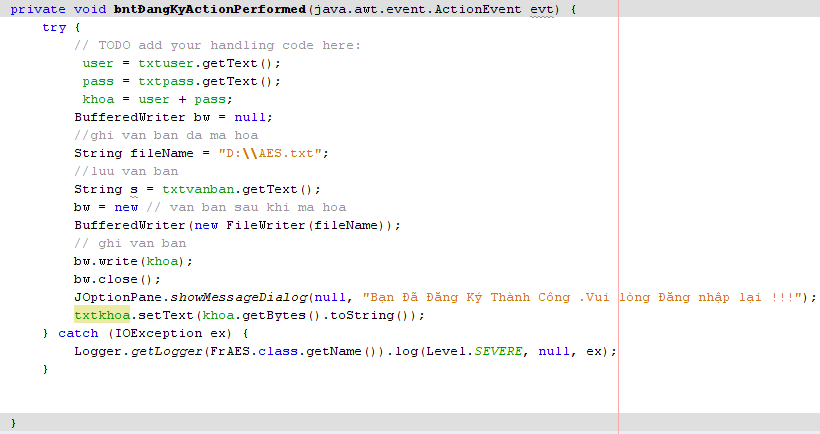


**Bước 2**: Viết hàm xử lý sự kiện

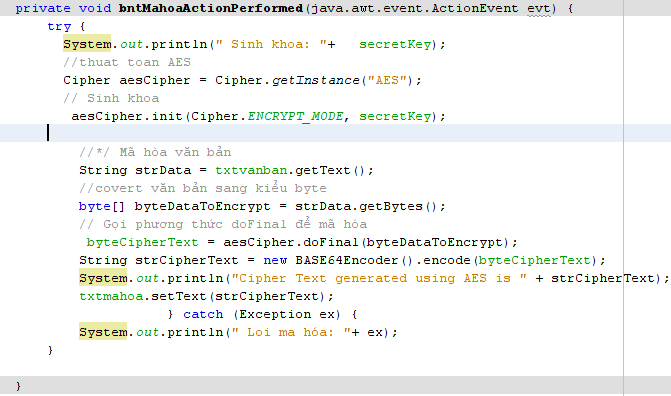
B2.1: Viết hàm xử lý sự kiện đăng nhập



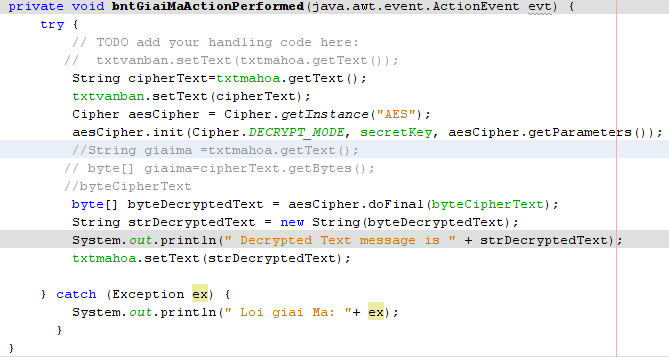
B2.2: Viết hàm xử lý sự kiện đăng ký



B2.3: Viết hàm xử lý sự kiện mã hóa



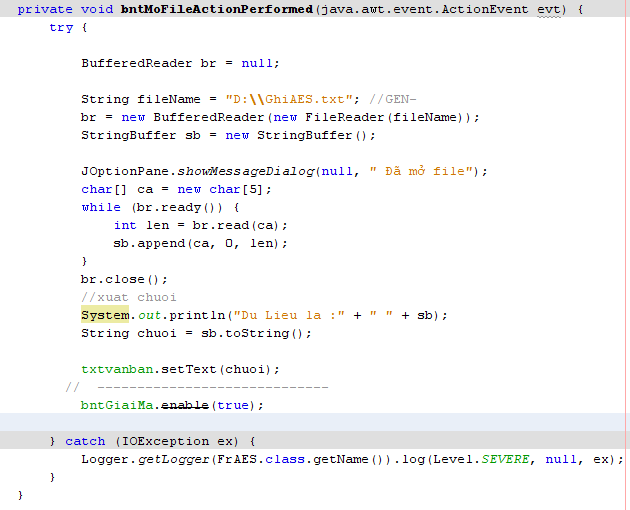
B2.3 Viết hàm xử lý sự kiện giải mã



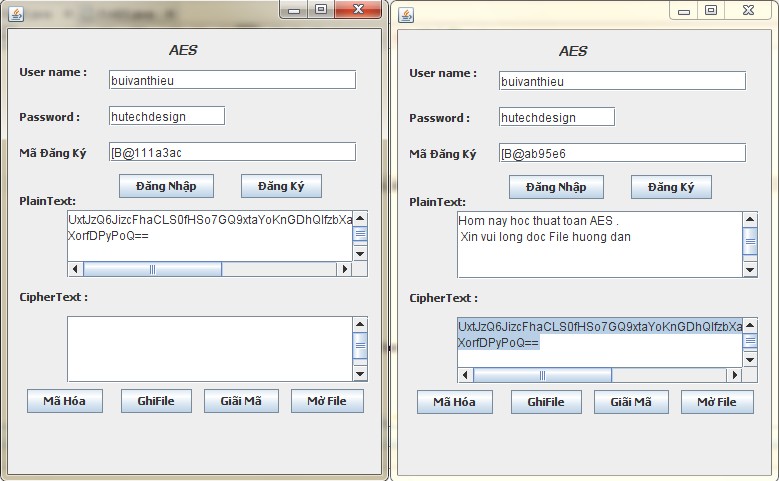
B2.4 Viết hàm xử lý sự kiện ghi File

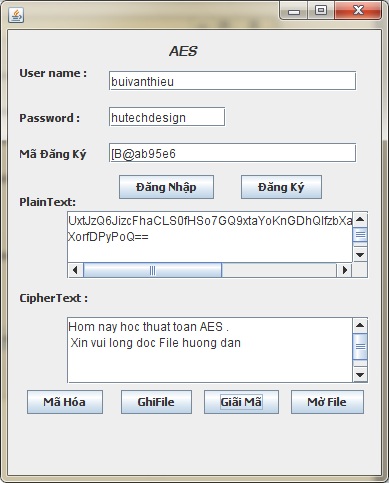


B2.5 Viết hàm xử lý sự kiện mở File



# Bước 3: Kiểm Tra





**Bài Tập: Viết phần mềm mã hóa văn bản với các thuật toán mã hóa trên. Chương trình có thể thực hiện các chức năng sau:**

**Cho phép nhập văn bản vào hệ thống.**

**Cho phép nhập khóa bảo vệ văn bản.**

**Cho phép mở File và Ghi File.**

**Cho phép bên gữi mã hóa dữ liệu và bên nhận mã hóa dữ liệu với khóa K. ( sinh viên nghiên cứu viết them chức năng..)**