**package** com.sanfoundry.setandstring;

**import** java.util.Scanner;

**public** **class** PlayfairCipherEncryption

{

**private** String KeyWord = **new** String();

**private** String Key = **new** String();

**private** **char** matrix\_arr[][] = **new** **char**[5][5];

/\*

\* method removes all duplicates from string k and creates new string adjustedKey

\*/

**public** **void** setKey(String k)

{

String adjustedKey = **new** String();

**boolean** sameChar = **false**;

adjustedKey = adjustedKey + k.charAt(0);

**for** (**int** i = 1; i < k.length(); i++)

{

**for** (**int** j = 0; j < adjustedKey.length(); j++)

{

**if** (k.charAt(i) == adjustedKey.charAt(j))

{

sameChar = **true**;

}

}

**if** (sameChar == **false**)

adjustedKey = adjustedKey + k.charAt(i);

sameChar = **false**;

}

KeyWord = adjustedKey;

}

/\*

\* method generates key by appending alphabet to key w/ no repeated letters

\*/

**public** **void** KeyGen()

{

**boolean** append = **true**;

**char** current;

Key = KeyWord;

**for** (**int** i = 0; i < 26; i++)

{

current = (**char**) (i + 97);

//exclude letter j

**if** (current == 'j')

**continue**;

**for** (**int** j = 0; j < KeyWord.length(); j++)

{

**if** (current == KeyWord.charAt(j))

{

append = **false**;

**break**;

}

}

**if** (append)

Key = Key + current;

append = **true**;

}

System.***out***.println(Key);

createCipherGrid();

}

/\*

\* creates 5x5 matrix grid with key

\*/

**private** **void** createCipherGrid()

{

**int** counter = 0;

**for** (**int** i = 0; i < 5; i++)

{

**for** (**int** j = 0; j < 5; j++)

{

matrix\_arr[i][j] = Key.charAt(counter);

System.***out***.print(matrix\_arr[i][j] + " ");

counter++;

}

System.***out***.println();

}

}

/\*

\* replace letters i with j and appends x to separate repeated letters

\*/

**private** String format(String old\_text)

{

**int** i = 0;

**int** len = 0;

String text = **new** String();

len = old\_text.length();

**for** (**int** tmp = 0; tmp < len; tmp++)

{

**if** (old\_text.charAt(tmp) == 'j')

{

text = text + 'i';

}

**else**

text = text + old\_text.charAt(tmp);

}

len = text.length();

**for** (i = 0; i < len; i = i + 2)

{

//separates repeated letters

**if** (text.charAt(i + 1) == text.charAt(i))

{

text = text.substring(0, i + 1) +

'x' + text.substring(i + 1);

}

}

**return** text;

}

/\*

\* appends x if string length is not even and puts pairs of letters into array x

\*/

**private** String[] Divid2Pairs(String new\_string)

{

String Original = format(new\_string);

**int** size = Original.length();

**if** (size % 2 != 0)

{

//appending x increases size

size++;

Original = Original + 'x';

}

String letterPairs[] = **new** String[size / 2];

**int** counter = 0;

**for** (**int** i = 0; i < size / 2; i++)

{

letterPairs[i] = Original.substring(counter, counter + 2);

counter = counter + 2;

}

**return** letterPairs;

}

/\*

\*gets position of each letter from 5x5 matrix

\*/

**public** **int**[] GetDiminsions(**char** letter)

{

**int**[] dimensions = **new** **int**[2];

**if** (letter == 'j')

letter = 'i';

**for** (**int** i = 0; i < 5; i++)

{

**for** (**int** j = 0; j < 5; j++)

{

**if** (matrix\_arr[i][j] == letter)

{

dimensions[0] = i;

dimensions[1] = j;

**break**;

}

}

}

**return** dimensions;

}

/\*

\*alters array to encode message

\*/

**public** String encryptMessage(String userInput)

{

String src\_arr[] = Divid2Pairs(userInput);

String Code = **new** String();

**char** firstLetter;

**char** secondLetter;

**int** Dimensions1st[] = **new** **int**[2];

**int** Dimensions2nd[] = **new** **int**[2];

**for** (**int** i = 0; i < src\_arr.length; i++)

{

firstLetter = src\_arr[i].charAt(0);

secondLetter = src\_arr[i].charAt(1);

Dimensions1st = GetDiminsions(firstLetter);

Dimensions2nd = GetDiminsions(secondLetter);

**if** (Dimensions1st[0] == Dimensions2nd[0])

{

**if** (Dimensions1st[1] < 4)

Dimensions1st[1]++;

**else**

Dimensions1st[1] = 0;

**if** (Dimensions2nd[1] < 4)

Dimensions2nd[1]++;

**else**

Dimensions2nd[1] = 0;

}

**else** **if** (Dimensions1st[1] == Dimensions2nd[1])

{

**if** (Dimensions1st[0] < 4)

Dimensions1st[0]++;

**else**

Dimensions1st[0] = 0;

**if** (Dimensions2nd[0] < 4)

Dimensions2nd[0]++;

**else**

Dimensions2nd[0] = 0;

}

**else**

{

**int** temp = Dimensions1st[1];

Dimensions1st[1] = Dimensions2nd[1];

Dimensions2nd[1] = temp;

}

Code = Code + matrix\_arr[Dimensions1st[0]][Dimensions1st[1]]

+ matrix\_arr[Dimensions2nd[0]][Dimensions2nd[1]];

}

**return** Code;

}

/\*

\* takes user input to be encoder, starts and calls methods

\*/

**public** **static** **void** main(String[] args)

{

PlayfairCipherEncryption message = **new**

PlayfairCipherEncryption();

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter a keyword:");

String keyword = sc.next();

message.setKey(keyword);

message.KeyGen();

System.***out***.println("Enter word to encrypt: " +

"(Make sure length of message is even)");

String userInput = sc.next();

//if (userInput.length() % 2 == 0)

//{

System.***out***.println("Encryption: " +

message.encryptMessage(userInput));

//}

//else

//{

// System.out.println("Message length should be even");

//}

sc.close();

}

}