Write a Java/C/C++/Python program to perform encryption and decryption using the method of Transposition technique.

**public** **class** TranspositionCipher

{

**public** **static** String *selectedKey*;

**public** **static** **char** *sortedKey*[];

**public** **static** **int** *sortedKeyPos*[];

// default constructor define the default key

**public** TranspositionCipher()

{

*selectedKey* = "megabuck";

*sortedKeyPos* = **new** **int**[*selectedKey*.length()];

*sortedKey* = *selectedKey*.toCharArray();

}

// Parameterized constructor define the custom key

**public** TranspositionCipher(String myKey)

{

*selectedKey* = myKey;

*sortedKeyPos* = **new** **int**[*selectedKey*.length()];

*sortedKey* = *selectedKey*.toCharArray();

}

// To reorder data do the sorting on selected key

**public** **static** **void** doProcessOnKey()

{

// Find position of each character in selected key and arrange it on

// alphabetical order

**int** min, i, j;

**char** orginalKey[] = *selectedKey*.toCharArray();

**char** temp;

// First Sort the array of selected key

**for** (i = 0; i < *selectedKey*.length(); i++)

{

min = i;

**for** (j = i; j < *selectedKey*.length(); j++)

{

**if** (*sortedKey*[min] > *sortedKey*[j])

{

min = j;

}

}

**if** (min != i)

{

temp = *sortedKey*[i];

*sortedKey*[i] = *sortedKey*[min];

*sortedKey*[min] = temp;

}

}

// Fill the position of array according to alphabetical order

**for** (i = 0; i < *selectedKey*.length(); i++)

{

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

{

**if** (orginalKey[i] == *sortedKey*[j])

*sortedKeyPos*[i] = j;

}

}

}

// to encrypt the targeted string

**public** **static** String doEncryption(String plainText)

{

**int** min, i, j;

**char** orginalKey[] = *selectedKey*.toCharArray();

**char** temp;

*doProcessOnKey*();

// Generate encrypted message by doing encryption using Transpotion

// Cipher

**int** row = plainText.length() / *selectedKey*.length();

**int** extrabit = plainText.length() % *selectedKey*.length();

**int** exrow = (extrabit == 0) ? 0 : 1;

**int** rowtemp = -1, coltemp = -1;

**int** totallen = (row + exrow) \* *selectedKey*.length();

**char** pmat[][] = **new** **char**[(row + exrow)][(*selectedKey*.length())];

**char** encry[] = **new** **char**[totallen];

**int** tempcnt = -1;

row = 0;

**for** (i = 0; i < totallen; i++)

{

coltemp++;

**if** (i < plainText.length())

{

**if** (coltemp == (*selectedKey*.length()))

{

row++;

coltemp = 0;

}

pmat[row][coltemp] = plainText.charAt(i);

}

**else**

{ // do the padding ...

pmat[row][coltemp] = '\*';

}

}

**int** len = -1, k;

**for** (i = 0; i < *selectedKey*.length(); i++)

{

**for** (k = 0; k < *selectedKey*.length(); k++)

{

**if** (i == *sortedKeyPos*[k])

{

**break**;

}

}

**for** (j = 0; j <= row; j++)

{

len++;

encry[len] = pmat[j][k];

}

}

String p1 = **new** String(encry);

**return** (**new** String(p1));

}

// to decrypt the targeted string

**public** **static** String doDecryption(String s)

{

**int** min, i, j, k;

**char** key[] = *selectedKey*.toCharArray();

**char** encry[] = s.toCharArray();

**char** temp;

*doProcessOnKey*();

// Now generating plain message

**int** row = s.length() / *selectedKey*.length();

**char** pmat[][] = **new** **char**[row][(*selectedKey*.length())];

**int** tempcnt = -1;

**for** (i = 0; i < *selectedKey*.length(); i++)

{

**for** (k = 0; k < *selectedKey*.length(); k++)

{

**if** (i == *sortedKeyPos*[k])

{

**break**;

}

}

**for** (j = 0; j < row; j++)

{

tempcnt++;

pmat[j][k] = encry[tempcnt];

}

}

// store matrix character in to a single string

**char** p1[] = **new** **char**[row \* *selectedKey*.length()];

k = 0;

**for** (i = 0; i < row; i++)

{

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

{

**if** (pmat[i][j] != '\*')

{

p1[k++] = pmat[i][j];

}

}

}

p1[k++] = '\0';

**return** (**new** String(p1));

}

@SuppressWarnings("static-access")

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

{

TranspositionCipher tc = **new** TranspositionCipher();

System.***out***.println("Encrypted Message is: "

+ tc.*doEncryption*("Sanfoundry"));

System.***out***.println("Decrypted Message is: "

+ tc.*doDecryption*(tc.*doEncryption*("Sanfoundry")));

}

}

Output:

Encrypted Message is: e d yacven\*hiaa

Decrypted Message is: have a nice day