МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра програмних систем і технологій

**Практична робота №2**

з дисципліни

**«Захист програм та даних»**

**Тема :** **«Шифри перестановки»**

Виконали:

Ємець Євгенія Ярославівна

Олійник Анастасія Миколаївна

Група ІПЗ-33

Перевірила:

Супрун Ольга Миколаївна

Дата перевірки

2018

**Хід роботи**

**Завдання №1**

Необхідно зашифрувати своє прізвище за допомогою шифру простої одинарної перестановки

Перестановка є спосіб шифрування, при якому для отримання шифрограми букви вихідного повідомлення міняють місцями. Типовим прикладом перестановки є анаграми, що стали популярними в XVII в. Анаграма (грец. Ανα - «знову» і γράμμα - «запис») - літературний прийом, що складається в перестановці букв або звуків певного слова (або словосполучення), що в результаті дає інше слово або словосполучення. Наприклад: апельсин - спанієль, полковник - клоповник, горілка - рогалик, пелюстка - телескоп.

Шифр простий одинарної перестановки. Для шифрування і дешифрування використовується таблиця перестановок, аналогічна показаної на рис.5.2.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 2 | 4 | 1 | 7 | 6 | 5 | 3 |

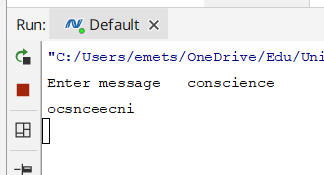
Рис.5.2. Таблица перестановок

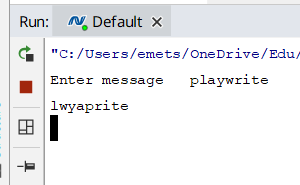
Наприклад, якщо для шифрування вихідного повідомлення «АБРАМОВ» використовувати таблицю, представлену на рис.5.2, то шифрограмою буде «РАВБОМА». Для використання на практиці такий шифр незручний, так як при великих значеннях n доводиться працювати з довгими таблицями і для повідомлень різної довжини необхідно мати свою таблицю перестановок.

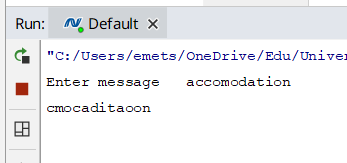
Лістинг програми

**using** System;  
**using** System.Collections.Generic;  
**using** System.Linq;  
  
**namespace** Laba2{  
 **internal class** Program{  
 **private static int state** = 1;  
 **private static readonly int**[] **K** = {2,5,4,3,1};  
 **private static readonly int**[] **Key** = **K**;  
   
 **private static int**[] Getunkey (IReadOnlyList<**int**> k){  
 **var** res = **new int**[k.**Count**];  
 **for** (**var** i = 0; i < k.**Count**; i++){  
 **var** ki = k[i];  
 ki--;  
 res[ki] = i + 1;  
 }  
 **return** res;  
 }  
   
 **private static string** Crypt (**string** msg,**bool** f){  
 **var** k1 = f ? **Key** : Getunkey (**Key**);  
 **var** res = **""**;  
   
 **for** (**var** i = 0; i < (1+(msg.**Length**/k1.**Length**)); i++){  
 **if** ((i + 1) \* k1.**Length** > msg.**Length**){  
 **var** r = i \* k1.**Length**;  
 r = msg.**Length** - r;  
 **return** msg.Substring(i \* k1.**Length**, r);  
 }  
  
 **var** tmp1 = msg.Substring (i \* k1.**Length**, k1.**Length**);  
 **var** tmp2 = k1.Select(t => t - 1).Aggregate(**""**, (current, ki) => current + tmp1.Substring(ki, 1));  
  
 res += tmp2;  
 }  
 **return** res;  
 }  
   
 **public static void** Main(**string**[] args){  
 Console.Write(**"Enter message\t"**);  
 **var** message = Console.ReadLine();  
 Console.WriteLine(Crypt(message, **true**));  
 Console.ReadKey(**true**);  
 }  
 }  
}

Приклади роботи програми







**Завдання №2**

Необхідно зашифрувати своє прізвище за допомогою блочної одинарної перестановки

Перестановочний шифр — алгоритм шифрування, який полягає у перестановці знаків відкритого тексту згідно з певним правилом, яке є ключем.

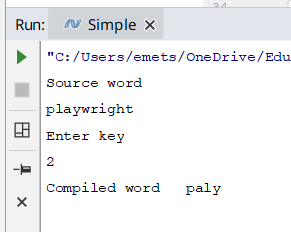
Наприклад, текст «знак», зашифрований ключем «3421», буде виглядати так: «казн».

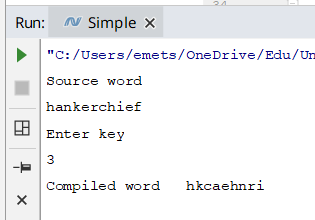
|  |  |  |  |
| --- | --- | --- | --- |
| з | н | а | к |
| 3 | 4 | 2 | 1 |
| \ | \ | / | / |
| / | / | \ | \ |
| 1 | 2 | 3 | 4 |
| к | а | з | н |

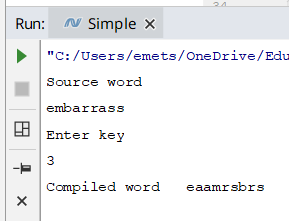
Лістинг програми

**using** System;  
**using static** System.String;  
  
**namespace** Simple{  
 **internal class** Program{  
 **public static void** Main(**string**[] args){  
 **var** massiv = **new char**[100,100];  
 **int** c, count = 0;  
   
 Console.WriteLine(**"Source word\t"**);  
 **var** s = Console.ReadLine()?.ToCharArray();  
 Console.WriteLine(**"Enter key\t"**);  
 **var** b = c = Convert.ToInt32(Console.ReadLine());  
   
 **for**(**var** i=0; i<b; i++){  
 **for**(**var** j=0; j<c; j++){  
 **if** (s != **null**) massiv[i, j] = s[count];  
 count++;  
 }  
 }  
   
 **var** result = **Empty**;  
 **for**(**var** j=0; j<c; j++){  
 **for**(**var** i=0; i<b; i++){  
 **if**(massiv[i, j] >= **'a'** && massiv[i, j] <= **'z'**)  
 result+=(massiv[i, j]);  
 }  
 }  
 Console.WriteLine(**"Compiled word\t{0}"**, result);  
 }  
 }  
}

Приклади роботи програми







**Завдання №3**

Необхідно зашифрувати своє прізвище за допомогою табличної маршрутної перестановки

При шифруванні в таку таблицю вписують вихідне повідомлення за певним маршрутом, а виписують (отримують шифрограму) - по іншому. Для даного шифру маршрути вписування і виписування, а також розміри таблиці є ключем.

Наприклад, вихідне повідомлення «АБРАМОВ ИЛЬЯ СЕРГЕЕВИЧ» вписується в прямокутну таблицю розмірами 4х6, маршрут вписування - зліва-направо зверху-вниз, маршрут виписування - зверху-вниз зліва-направо. Шифрограма в цьому випадку виглядає «АВ\_ЕБ\_СВРІЕІАЛР ЧМЬГ\_ОЯЕ\_».

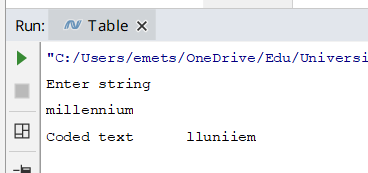
Шифр вертикальної перестановки. Є різновидом попереднього шифру. До особливостей шифру можна віднести наступні:

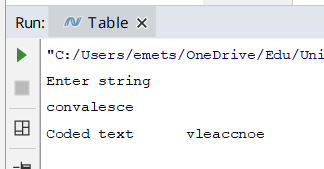
* кількість стовпців в таблиці фіксується і визначається довжиною ключа;
* маршрут вписування строго відповідає маршруту, показаному на рис.12;
* шифрограма виписується за стовпцями відповідно до їх нумерацією (ключем).

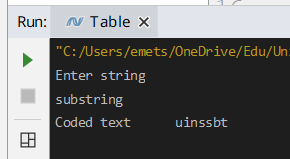
Лістинг програми

**using** System;  
**using** System.Linq;  
**using static** System.String;  
  
**namespace** Table{  
 **internal class** Program{  
 **private static string** Encrypt(**string** text, **int**[] key){  
 **var** newText = **Empty**;  
 **for** (**var** i = 0; i < text.**Length**-1; i += 1){  
 **var** index = key[i] - 1;  
 **if** (i >= 0 && index >= 0) newText += text[index];  
 }  
 **return** newText;  
 }  
   
 **public static void** Main(**string**[] args){  
 Console.WriteLine(**"Enter string\t"**);  
 **var** text = Console.ReadLine();  
 **if** (text == **null**) **return**;  
 **var** key = **new int**[text.**Length**];  
  
 **for** (**var** i = 0; i < text.**Length**; i += 1)  
 key[i] = i;  
 **var** rnd = **new** Random();  
 key = key.OrderBy(x => rnd.Next()).ToArray();  
   
 Console.WriteLine(**"Coded text\t{0}"**, Encrypt(text, key));  
 }  
 }  
}

Приклади роботи програми







**Завдання №4**

Необхідно зашифрувати своє прізвище за допомогою вертикальної перестановки

Широке поширення отримала різновид маршрутної перестановки - вертикальна перестановка. У цьому шифрі також використовується прямокутна таблиця, в яку повідомлення записується по рядках зліва направо. Виписується шифрограма по вертикалях, при цьому стовпці вибираються в порядку, визначеному ключем.

Відкритий текст: пример маршрутной перестановки

Ключ: (3, 1, 4, 2, 5)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3 | 1 | 4 | 2 | 5 |
| п | р | и | м | е |
| р | м | а | р | ш |
| р | у | т | н | о |
| й | п | е | р | е |
| с | т | а | н | о |
| в | к | и |  |  |

Криптограмма: рмупткмрнрнпррйсвиатеаиешоео

Заповнювати останній рядок таблиці «неробочими» буквами недоцільно, так як криптоаналитик, який отримав цю криптограму, отримує відомості про довжину числового ключа.

Лістинг програми

**using** System;  
   
**namespace** Vertical{  
 **internal class** Vertical{  
 **public static void** Main(**string**[] args){  
 **var** text = Console.ReadLine();  
 **if** (text != **null**){  
 **var** k = text.**Length**;  
 }  
  
 **var** array = **new string**[7, 9];  
 **var** q = 0;  
 **for** (**var** i = 0; i < 7; i++)  
 **for** (**var** j = 0; j < 7; j++){  
 **if** (text != **null** && q < text.**Length**)  
 array[i, j] = Convert.ToString(text[q++]);  
 **else if** (text != **null** && (q <= text.**Length** && j == 0))  
 **break**;  
 **else** array[i, j] = **"-"**;  
 }  
  
 **var** tmp = **string**.**Empty**;  
 **for** (**var** i = 0; i < 7; i++){  
 **for** (**var** j = 0; j < 7; j++)  
 tmp += array[i, j];  
 tmp+=**"\n"**;  
 }  
   
 **var** result = **""**;  
 **var** key = tmp;  
 **for** (**var** i = 0; i < 7; i++)  
 **for** (**var** j = 0; j < 7; j++)  
 **if** ( i== Convert.ToInt32(Convert.ToString(key[j])))  
 **for** (**var** t = 0; t < 7; t++)  
 result += array[t, j];  
   
 Console.WriteLine(result);  
 Console.ReadKey(**true**);  
 }  
 }  
}

**Завдання №5**

Необхідно зашифрувати своє прізвище за допомогою поворотної решітки

У 1550 році італійський математик Джироламо Кардано (1501-1576) в книзі «Про тонкощі» запропонував нову техніку шифрування повідомлень - грати.

Спочатку решітка Кардано представляла собою трафарет з отворами, в які записували букви, склади або слова повідомлення. Потім трафарет прибирали, а вільне місце заповнювали більш-менш осмисленим текстом. Такий метод приховування інформації відноситься до стеганографії.

Пізніше був запропонований шифр «поворотна решітка» - перший транспозіціонний (геометричний) шифр. Незважаючи на те, що існує велика різниця між початковим пропозицією Кардано і шифром «поворотна решітка», методи шифрування, засновані на трафаретах, прийнято називати «гратами Кардано».

Для шифрування і дешифрування за допомогою даного шифру виготовляється трафарет з вирізаними отворами. При накладенні трафарету на таблицю того ж розміру чотирма можливими способами, його вирізи повністю повинні покривати всі клітини таблиці рівно по одному разу.

При шифруванні трафарет накладають на таблицю. В видимі осередки по певному маршруту вписують літери відкритого тексту. Далі трафарет перевертають три рази, кожен раз роблячи операцію заповнення.

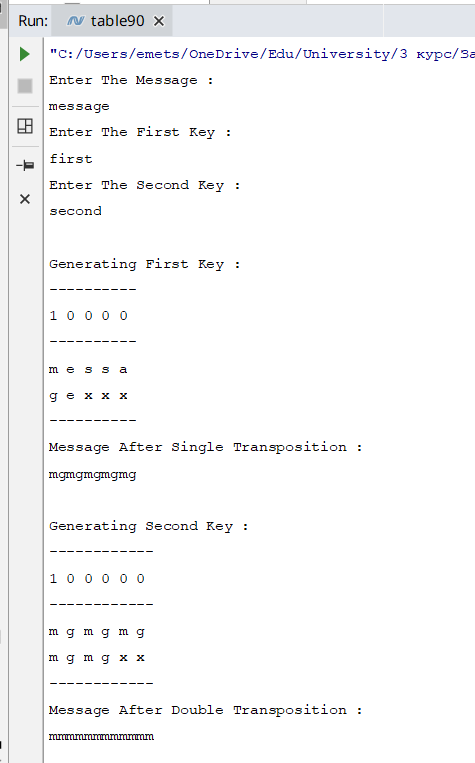
Шифрограму виписують з отриманої таблиці по певному маршруту. Ключем є трафарет, маршрут вписування і порядок поворотів.

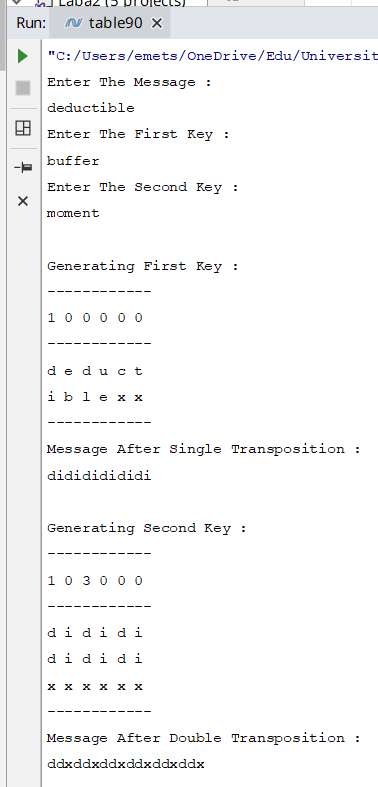
Даний метод шифрування використовувався для передачі секретної інформації нідерландськими правителями в 1740-х роках. Під час Першої світової війни армія кайзера Вільгельма використовувала шифр «поворотна решітка». Німці використовували решітки різних розмірів, проте дуже недовго (чотири місяці), до величезного розчарування французьких криптоаналітиків, які тільки-тільки почали підбирати до них ключі. Для решіток різних розмірів французи придумали власні кодові імена: Анна (25 букв), Берта (36 букв), Дора (64 літери) та Еміль (81 буква).

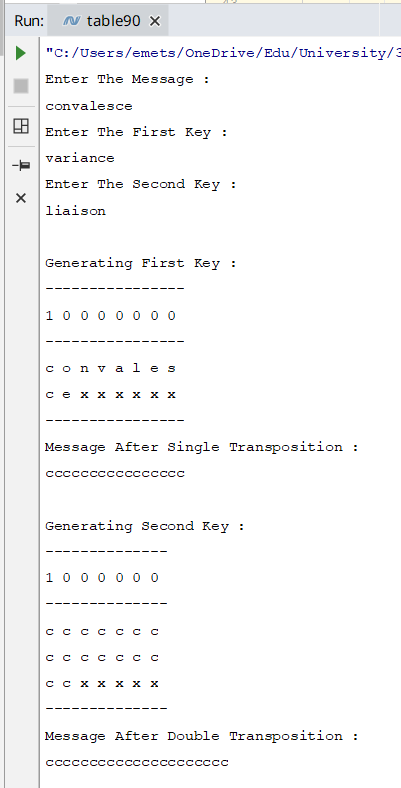
Лістинг програми

**using** System;  
**using** System.Linq;  
  
**namespace** table90  
{  
 **public class** DoTran  
 {  
 **public readonly string Msg**;  
 **public readonly string Key1**;  
 **public readonly string Key2**;  
 **public string Enc1**;  
 **public string Enc2** { **get**; **set**; }  
 **public string** Decy2;  
 **public readonly int**[] **Ord1**;  
 **public readonly int**[] **Ord2**;  
 **public readonly int L1**;  
 **public readonly int l2**;  
  
 **public** DoTran(**string** s, **string** k1, **string** k2)  
 {  
 **Msg** = s;  
 **Key1** = k1;  
 **Key2** = k2;  
 **L1** = k1.**Length**;  
 **l2** = k2.**Length**;  
 **Ord1** = **new int**[**L1**];  
 **Ord2** = **new int**[**l2**];  
 }  
  
 **public void** genKey(**string** s, **int**[] p)  
 {  
 **int** k = 1;  
 **bool**[] v = **new bool**[s.**Length**];  
 **for** (**int** i = 0;i < s.**Length**;i++)  
 {  
 v[i] = **false**;  
 }  
  
 **string** t = s;  
 s.OrderBy(x => s.GetEnumerator());  
 **foreach** (**var** t1 **in** s)  
 {  
 **for** (**var** j = 0;j < s.**Length**;j++)  
 {  
 **if** (t1 == t[j] && v[j] == **false**)  
 {  
 p[j] = k;  
 }  
   
 v[j] = **true**;  
 k++;}  
 }  
 Console.Write(**"\n"**);  
 }  
 **public string** encrypt(**string** s, **int**[] ord, **int** k)  
 {  
 **string** str = String.**Empty**;  
 **int** c = 1;  
 **int** f = 0;  
 **int** l = s.**Length** / k;  
 **char** p;  
 **for** (**int** i = 0;i < k;i++)  
 {  
 **for** (**int** j = 0;j < k;j++)  
 {  
 **if** (c == ord[j])  
 {  
 f = j;  
 }  
   
 j = k;  
 c++;  
 }  
 **for** (**int** j = 0;j < l + 1;j++)  
 {  
 **if** (j \* k + f < s.**Length**)  
 {  
 p = s[j \* k + f];  
 }  
 **else** {  
 p = **'x'**;  
 }  
  
 str = str + p;  
 }  
 }  
 Console.Write(str);  
 Console.Write(**"\n"**);  
 **return** str;  
 }  
 **public void** printMat(**string** s, **int**[] ord, **int** l)  
 {  
 **int** z = s.**Length** / l;  
 z = (z + 1) \* l;  
 **for** (**int** i = 0;i < l;i++)  
 {  
 Console.Write(**"--"**);  
 }  
  
 Console.Write(**"\n"**);  
 **for** (**int** i = 0;i < l;i++)  
 {  
 Console.Write(ord[i]);  
 Console.Write(**" "**);  
 }  
  
 Console.Write(**"\n"**);  
 **for** (**int** i = 0;i < l;i++)  
 {  
 Console.Write(**"--"**);  
 }  
  
 Console.Write(**"\n"**);  
 **for** (**int** i = 0;i < z;i++)  
 {  
 **if** (i < s.**Length**)  
 {  
 Console.Write(s[i]);  
 }  
 **else** {  
 Console.Write(**"x"**);  
 }  
 **if** ((i + 1) % l == 0)  
 {  
 Console.Write(**"\n"**);  
 }  
 **else** {  
 Console.Write(**" "**);  
 }  
 }  
 **for** (**int** i = 0;i < l;i++)  
 {  
 Console.Write(**"--"**);  
 }  
 Console.Write(**"\n"**);  
 }  
 }  
  
 **public static class** GlobalMembers  
 {  
 **static void** Main()  
 {  
 Console.Write(**"Enter The Message : "**);  
 Console.Write(**"\n"**);  
 **var** s = Console.ReadLine();  
 Console.Write(**"Enter The First Key : "**);  
 Console.Write(**"\n"**);  
 **var** k1 = Console.ReadLine();  
 Console.Write(**"Enter The Second Key : "**);  
 Console.Write(**"\n"**);  
 **var** k2 = Console.ReadLine();  
 DoTran d = **new** DoTran(s,k1,k2);  
  
 Console.Write(**"\nGenerating First Key : "**);  
 d.genKey(d.**Key1**,d.**Ord1**);  
 d.printMat(d.**Msg**,d.**Ord1**,d.**L1**);  
 Console.Write(**"Message After Single Transposition : "**);  
 Console.Write(**"\n"**);  
 d.**Enc1** = d.encrypt(d.**Msg**,d.**Ord1**,d.**L1**);  
 Console.Write(**"\nGenerating Second Key : "**);  
 d.genKey(d.**Key2**,d.**Ord2**);  
 d.printMat(d.**Enc1**,d.**Ord2**,d.**l2**);  
  
 Console.Write(**"Message After Double Transposition : "**);  
 Console.Write(**"\n"**);  
 d.**Enc2** = d.encrypt(d.**Enc1**,d.**Ord2**,d.**l2**);  
  
 }  
 }  
}

Приклади роботи програми







**Завдання №6**

Необхідно зашифрувати своє прізвище за допомогою магічного квадрату

Магічний квадрат — це квадратна таблиця , заповнена числами таким чином, що сума чисел у кожному рядку, кожному стовпчику і на обох діагоналях однакова. Якщо в квадраті рівні суми чисел тільки в рядках і стовпцях, то він називається напівмагічним. Нормальним називається магічний квадрат, заповнений цілими числами від 1 до . Магічний квадрат називається асоціативним або симетричним, якщо сума будь-яких двох чисел, розташованих симетрично щодо центру квадрата, дорівнює .

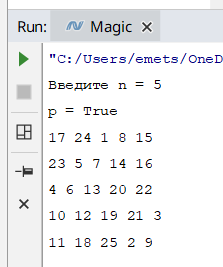
Нормальні магічні квадрати існують для всіх порядків , за винятком , хоча випадок тривіальний — квадрат складається з одного числа. Мінімальний нетривіальний випадок показаний нижче, він має порядок 3.

Сума чисел в кожному рядку, стовпчику і по діагоналях, називається магічною сталою, M. Магічна константа нормального магічного квадрата залежить тільки від n і визначається формулою:

Лістинг програми

**using** System;  
  
**namespace** Magic{  
 **internal class** Program{  
 **static void** Swap (**int** a, **int** b){  
 **var** T = a;  
 a = b;  
 b = T;  
 }  
  
 **static void** Main(**string**[] args){   
 **var** p = **true**;  
 **int** i, j, k;  
 **int** b, r, m;  
  
 Console.Write(**"Введите n = "**);  
 **var** n = Convert.ToInt32(Console.ReadLine());  
   
 **var** a = **new int**[n+1,n+1];  
  
 **if** (n % 2 != 0){ *// It's odd* i = 1;  
 j = (n/2) + 1; *// there is no Succ function in C#* **for** (k = 1; k <= n\*n; k++){  
 a[i,j] = k;  
 **if** ((k % n) == 0) i++;  
 **else**{  
 i--;  
 j++;  
 **if** (i == 0) i = n;  
 **if** (j > n) j = 1;  
 }  
 }  
 }  
 **else**{ *// It's even* **int** s;  
 **if** ((n % 4) == 0){  
 k = 1;  
 **for** (i = 1; i <= n; j++){  
 **for** (j = 1; j <= n; j++){  
 a[i,j] = k;  
 k++;  
 }  
 }  
 j = 2; m = n / 2;  
 **for** (i = 1; i <= m; i++){  
 **for** (k = 1; k <= m / 2; k++){  
 **if** (j == m + 1) j = 2;  
 **else**{  
 **if** (j == m + 2) j = 1;  
 s = (n - i) + 1;  
 b = (n - j) + 1;  
 Swap(a[i,j], a[s,b]);  
 Swap(a[i,b], a[s,j]);  
 j = j + 2;  
 }  
 }  
 }  
 }  
 **else**{  
 **if** (n != 2){  
 k = 1;  
 **for** (i = 1; i <= n; i++){  
 **for** (j = 1; j <= n; j++){  
 a[i,j] = k;  
 k++;  
 }  
  
 r = ((n/2-1))/2;   
 m = n/2;  
  
 **for** (i = 1; i <= m; i++){  
 j = i;  
  
 **for** (k = 1; k <= r; k++){  
 **if** (j > m) j = 1;  
 s = (n - i)+1;   
 b = (n - j)+1;  
 Swap(a[i,j], a[s,b]);  
 Swap(a[i,b], a[s,j]);  
 j++;  
 }  
 }  
   
 i = 1;   
 j = r+1;  
  
 **for** ( k = 1; k <= m; k++){  
 **if** (j > m) j = 1;  
 s = (n - i)+1;  
 Swap(a[i, j], a[s, j]);  
 i++;  
 j++;  
 }  
   
 i = 1;  
 j = r + 2;  
 **for** ( k = 1; k <= m; k++){  
 **if** (j > m) j = 1;  
 b = (n - j)+1;  
 Swap(a[i, j], a[i, b]);  
 i++;  
 j++;  
 }  
 }  
 }  
 **else** p = **false**;  
 }  
 }  
  
 **if** (p){   
 Console.Write(**"p = "**);  
 Console.Write(p);  
 Console.WriteLine();  
  
 **for** (i = 1; i <= n; i++){  
 **for** (j = 1; j <= n; j++){  
 Console.Write(a[i, j]);  
 Console.Write(**" "**);  
 }  
 Console.WriteLine();  
 }  
 }  
 **else** Console.WriteLine(**"do not exists"**);  
 }  
 }  
}

Приклад роботи програми



**Завдання №7**

Необхідно зашифрувати своє прізвище за допомогою подвійної перестановки

Шифрувальні таблиці з подвійною перестановкою по ключу використовують для підвищення скритності шифру. В даному методі використовуються два ключових слова. Перше слово визначає перестановку стовпців, друге - перестановку рядків таблиці. Перестановки виробляються згідно з порядком проходження в алфавіті символів ключових слів.

На першому етапі вихідний текст (або його фрагмент) через підрядник записується в таблицю. Далі переставляється стовпці вихідної таблиці по першому ключовому слову. Потім переставляються рядки отриманої таблиці по другому ключовим словом. На останньому етапі з підсумкової таблиці зчитується шифртекст за стовпцями.

Приклад 1. Зашифруємо фразу з третього прикладу за допомогою таблиці розміром 4х6 і ключових слів «СКАНЕР» і «4123».

Після заповнення вихідної таблиці по рядках (рис. 5а) переставляємо стовпці по порядку проходження в алфавіті букв слова «СКАНЕР» (рис. 5б). Потім переставляємо рядки. Порядковий номер рядка визначає цифра другого ключового слова «4123» (рис. 5в). На цьому перестановки в таблиці закінчуються. Шифртекст зчитуємо по стовпцях і отримуємо: «ЙЛЕСП\_ЕЕИОМІ\_ЬНТАІНМНРЗС»

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| А) вихідна таблиця   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  | С | К | А | Н | Е | Р | |  | 6 | 3 | 1 | 4 | 2 | 5 | | 4 | С | И | С | Т | Е | М | | 1 | Н | Ы | Й |  | П | А | | 2 | Р | О | Л | Ь |  | И | | 3 | З | М | Е | Н | Е | Н | | Б) перестановка стовпців   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  | А | Е | К | Н | Р | С | |  | 1 | 2 | 3 | 4 | 5 | 6 | | 4 | С | Е | И | Т | М | С | | 1 | Й | П | Ы |  | А | Н | | 2 | Л |  | О | Ь | И | Р | | 3 | Е | Е | М | Н | Н | З | | В) перестановка рядків   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  | А | Е | К | Н | Р | С | |  | 1 | 2 | 3 | 4 | 5 | 6 | | 1 | Й | П | Ы |  | А | Н | | 2 | Л |  | О | Ь | И | Р | | 3 | Е | Е | М | Н | Н | З | | 4 | С | Е | И | Т | М | С | |

Лістинг програми

**using** System;  
**using** System.Collections.Generic;  
**using** System.Linq;  
   
**namespace** TestConsole{   
 **internal class** CharNum{  
 **private char \_ch**;  
 **private int \_numberInWord**;  
   
 **public char Ch**{  
 **get** => **\_ch**;  
 **set**{  
 **if** (**\_ch** == **value**)  
 **return**;  
 **\_ch** = **value**;  
 }  
 }  
 **public int NumberInWord**{  
 **get** => **\_numberInWord**;  
 **set**{  
 **if** (**\_numberInWord** == **value**)  
 **return**;  
 **\_numberInWord** = **value**;  
 }  
 }  
 }  
  
 **static class** Program  
 {  
 **private static int** GetNumberInThealphabet(**char** s){  
 **const string *str*** = **@"АаБбВвГгДдЕеЁёЖжЗзИиЙйКкЛлМмНнОоПпРрСсТтУуФфХхЦцЧчШшЩщЪъЫыЬьЭэЮюЯя"**;  
 **var** number = ***str***.IndexOf(s) / 2;  
 **return** number;  
 }  
   
 **private static** List<CharNum> FillListKey(**char**[] chars){  
 **var** listKey = **new** List<CharNum>(chars.**Length**);  
 listKey.AddRange(chars.Select(t => **new** CharNum(){  
 **Ch** = t,  
 **NumberInWord** = GetNumberInThealphabet(t)  
 }));  
  
 **return** listKey;  
 }  
   
 **private static void** ShowKey(List<CharNum> listCharNum, **string** message){  
 Console.WriteLine(message);  
   
 **foreach** (**var** i **in** listCharNum)  
 Console.Write(i.**Ch** + **" "**);  
   
 Console.WriteLine();  
   
 **foreach** (**var** i **in** listCharNum)  
 Console.Write(i.**NumberInWord** + **" "**);  
   
 Console.WriteLine();  
 Console.WriteLine();  
 }  
 **private static** List<CharNum> FillingSerialsNumber(List<CharNum> listCharNum){  
 **var** count = 0;  
 **var** result = listCharNum.OrderBy(a => a.**NumberInWord**);  
   
 **foreach** (**var** i **in** result)  
 i.**NumberInWord** = count++;  
   
 **return** listCharNum;  
 }  
  
 **private static void** ShowMatrix(**char**[,] matrix, **string** message)  
 {  
 Console.WriteLine(message);  
 **for**(**var** i = 0; i < matrix.GetLength(0); i++){  
 **for**(**var** j = 0; j < matrix.GetLength(1); j++){  
 Console.Write(matrix[i, j] + **" "**);  
 }  
 Console.WriteLine();  
 }  
 Console.WriteLine();  
 Console.WriteLine();  
 }  
  
 **private static void** Main()  
 {   
 *// Первый ключ, количество столбцов* **const string *firstKey*** = **"Сканер"**;  
 *// Второй ключ, количество строк* **const string *secondKey*** = **"Надо"**;  
 *// Предложение которое шифруем* **const string *stringUser*** = **"Системный пароль изменен"**;  
   
 *// Матрица в которой производим шифрование* **var** matrix = **new char**[***secondKey***.**Length**, ***firstKey***.**Length**];  
   
 *// Счетчик символов в строке* **var** countSymbols = 0;  
   
 *// Переводим строки в массивы типа char* **var** charsFirstKey = ***firstKey***.ToCharArray();  
 **var** charsSecondKey = ***secondKey***.ToCharArray();  
 **var** charStringUser = ***stringUser***.ToCharArray();  
   
 *// Создаем списки в которых будут храниться символы и порядковы номера символов  
  
 // Заполняем символами из ключей* **var** listCharNumFirst = FillListKey(charsFirstKey);  
 **var** listCharNumSecond = FillListKey(charsSecondKey);  
   
 *// Заполняем порядковыми номерами* listCharNumFirst = FillingSerialsNumber(listCharNumFirst);  
 listCharNumSecond = FillingSerialsNumber(listCharNumSecond);  
   
 ShowKey(listCharNumFirst, **"Первый ключ: "**);  
 ShowKey(listCharNumSecond, **"Второй ключ: "**);  
   
 *// Заполнение матрицы строкой пользователя* **for** (**var** i = 0; i < listCharNumSecond.**Count**; i++){  
 **for**(**var** j = 0; j < listCharNumFirst.**Count**; j++){  
 matrix[i, j] = charStringUser[countSymbols++];  
 }  
 }  
   
 ShowMatrix(matrix, **"Первоначальное значение: "**);  
   
 countSymbols = 0;  
 *// Заполнение матрицы с учетом шифрования.   
 // Переставляем столбцы по порядку следования в первом ключе.   
 // Затем переставляем строки по порядку следования во втором ключа.* **foreach** (**var** t **in** listCharNumSecond){  
 **foreach** (**var** t1 **in** listCharNumFirst){  
 matrix[t.**NumberInWord**,  
 t1.**NumberInWord**] = charStringUser[countSymbols++];  
 }  
 }  
 ShowMatrix(matrix, **"Зашифрованное значение: "**);  
 Console.ReadKey();  
 }  
 }  
}

Приклад роботи програми

