

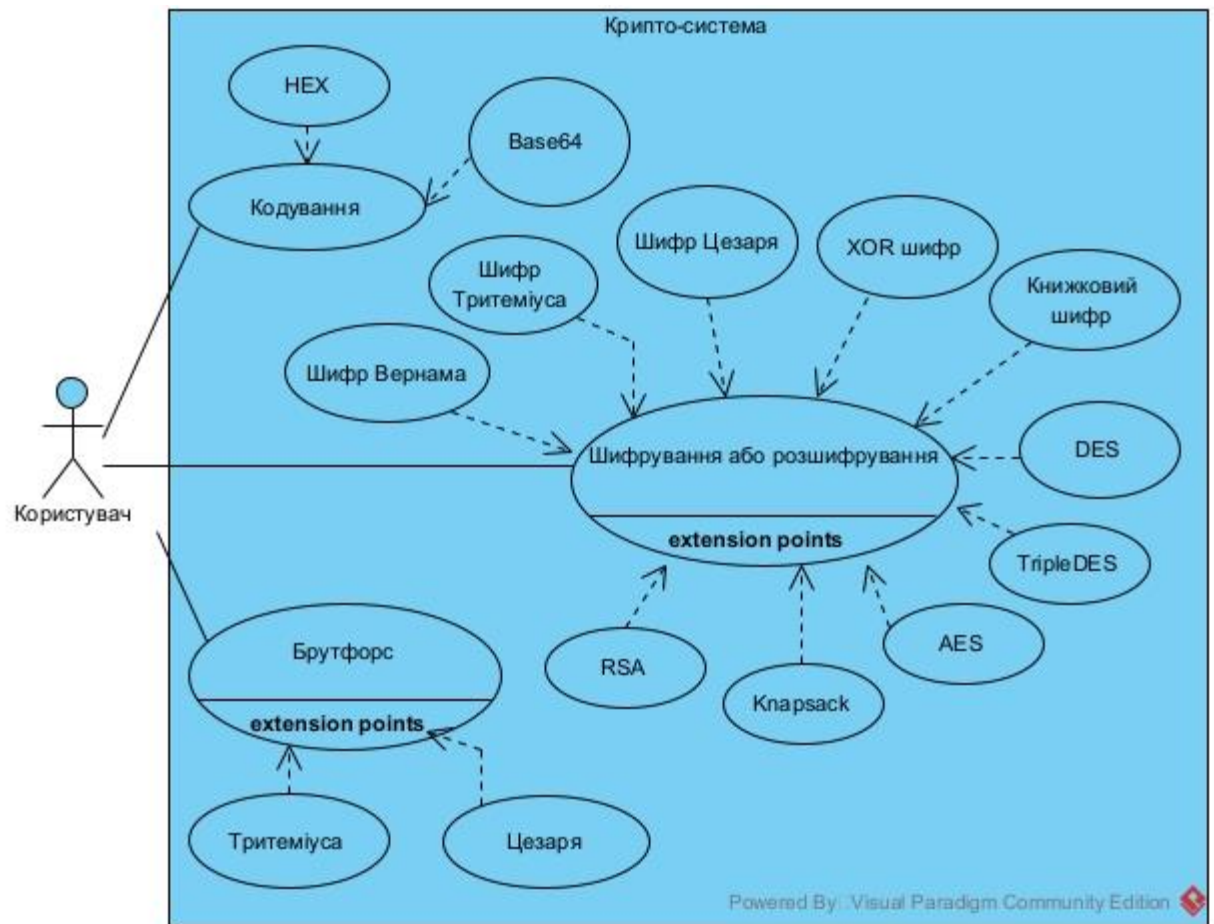
Міністерство освіти і науки України  
НТУУ «КПІ ім. Ігоря Сікорського»  
Навчально-науковий інститут атомної та теплової енергетики  
Кафедра цифрових технологій в енергетиці

Лабораторна робота №7  
з дисципліни «Безпека інформаційних систем»  
«Шифрування з відкритим ключем на основі алгоритму  
RSA»  
Варіант № 22

Виконав: Студент групи ТР-12  
Ковальов Олександр  
Перевірів: доцент, к.ф.-м.н.  
Тарнавський Ю. А.

**Мета роботи.** Ознайомитись з використанням криптопровайдерів .NET для побудови асиметричної криптосистеми

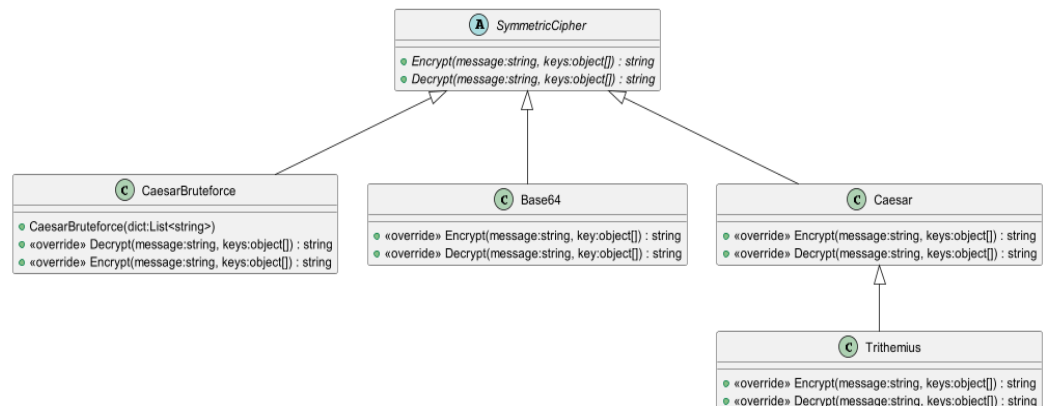
### Діаграма прецедентів.



### Діаграма класів.

В просторі імен Cryptography знаходяться всі шифри та супутні до них алгоритми. Також, там є перелік CipherEnum. Клас LocalRSA знаходиться в просторі імен Cryptography.Asymmetric.

В класі LocalRSA знаходяться основні методи для шифрування та розшифрування даних цим методом. АРІ класу складається з двох основних методів – Encrypt та Decrypt. В них викликаються приватні методи. Наприклад, там ще є методи для генерації секретного та публічного ключа.



## Фрагмент коду з реалізацією алгоритму шифрування/розшифрування.

```
public class LocalRSA
{
    private readonly RSAEncryptionPadding _padding;

    private readonly RSAEncryptionPadding DefaultPadding = RSAEncryptionPadding.OaepSHA256;

    public LocalRSA(RSAEncryptionPadding? padding = null)
    {
        _padding = padding ?? DefaultPadding;
    }

    public string Encrypt(string message, string publicKey)
    {
        using var provider = RSA.Create();

        provider.ImportFromPem(publicKey);

        var messageBytes = Encoding.Unicode.GetBytes(message);
        var encryptedBytes = provider.Encrypt(messageBytes, _padding);

        return Convert.ToBase64String(encryptedBytes);
    }

    public string Decrypt(string encryptedMessage, string privateKey)
    {
        using var provider = RSA.Create();

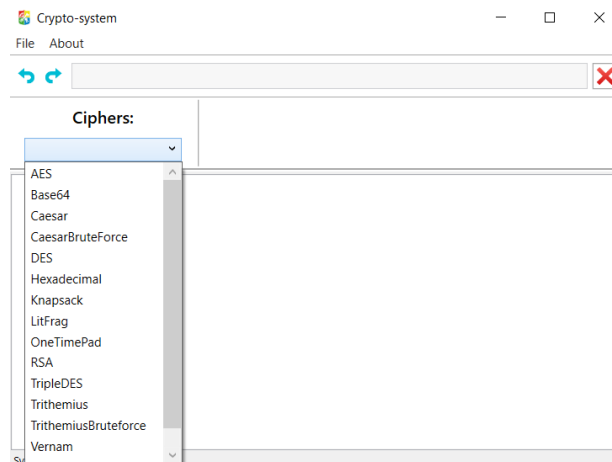
        provider.ImportFromPem(privateKey);

        var encryptedBytes = Convert.FromBase64String(encryptedMessage);
        var decryptedBytes = provider.Decrypt(encryptedBytes, _padding);

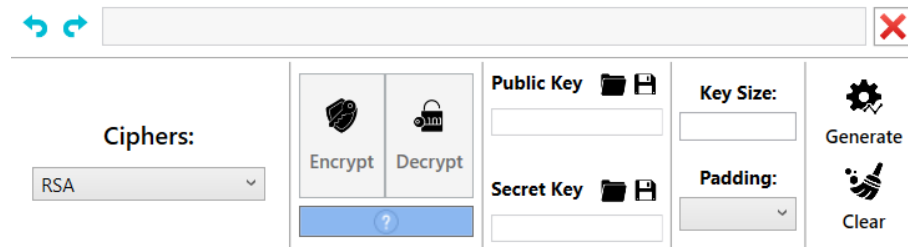
        return Encoding.Unicode.GetString(decryptedBytes);
    }
}
```

## Скріншоти програми.

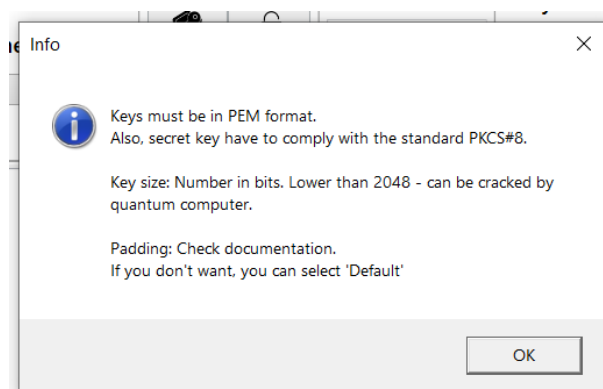
Головне вікно:



**RSA.** Головна панель має такий вигляд:



Якщо користувач не знає як користуватись шифром, то може натиснути на кнопку “Info” – вона знаходиться під кнопками для шифрування. Там він може дізнатись, які дані потрібно вводити в поля.



Є 4 поля для введення, одне з яких є випадаючим списком. Перше – це StatusBar з публічним ключем. Для розрізнення різних ключів в ньому показуються перші 10 символів захешованого повідомлення стандартом SHA1. Ключ повинен бути в PEM форматі. Також, секретний ключ повинен відповідати стандарту зберігання приватних ключів PKCS#8. Ключі можна або відкрити з відповідного файлу, або зберегти.

Окрім цього, наявне поле для введення розміру ключа. Це потрібно лише для його генерації. Бажаний розмір – 2048 або 4096+ біт.

Також, є список для вибору типу доповнення: в наявності PKCS#1, SHA1, SHA256, SHA384, SHA512. За замовчуванням – SHA256.

Кнопки Generate та Clear потрібні для генерації ключа за розміром та очищення полів відповідно.

<b>Public Key</b>	<b>Key Size:</b>	 Generate  Clear
<input type="text"/>	<input type="text"/>	
<b>Secret Key</b>	<b>Padding:</b>	
<input type="text"/>	<input type="text"/>	

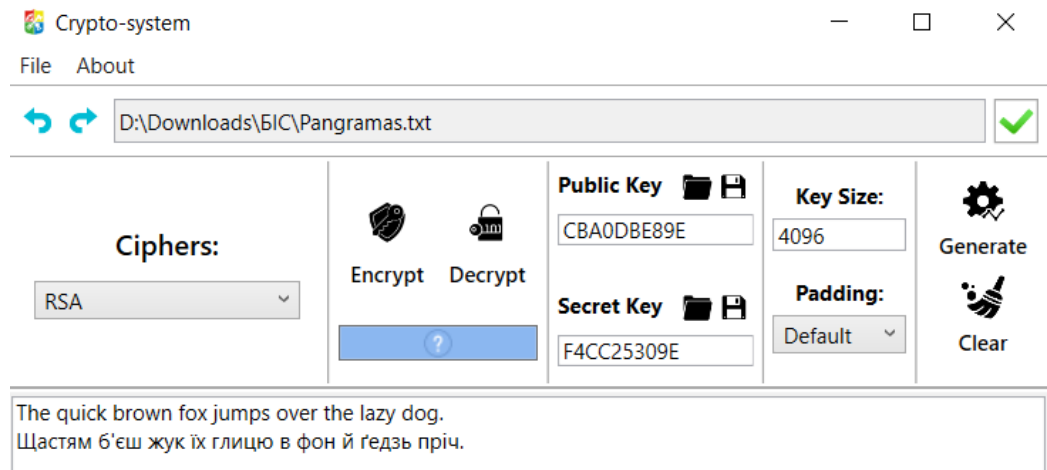
Приклад зашифрованого повідомлення:

D:\Downloads\5IC\Pangramas.txt

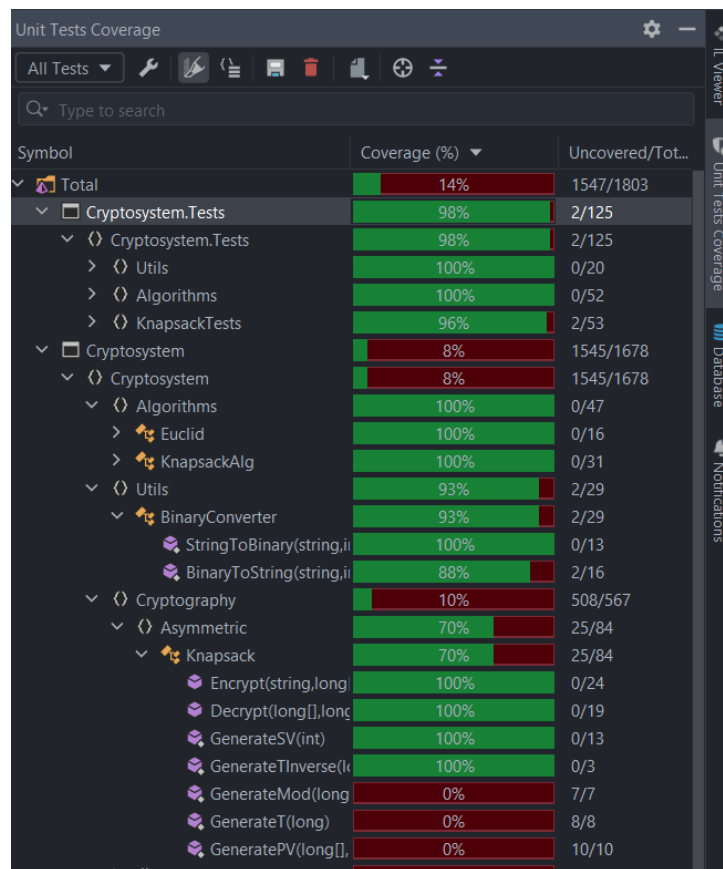
<b>Ciphers:</b> RSA	 Encrypt Decrypt 	<b>Public Key</b>	<b>Key Size:</b>	 Generate  Clear
		<input type="text" value="CBA0DBE89E"/>	<input type="text" value="4096"/>	
		<b>Secret Key</b>	<b>Padding:</b>	
		<input type="text" value="F4CC25309E"/>	<input type="text" value="Default"/>	

```
nwFHTeZndALSve5T/ZFBYhrT9N67tYq0YckP7W4HFZFzj3hYfh1+B/  
jNPBxtt4mm6QYBqd33Gjk37arvjWfzPb6yr45Jj3FalOhJGkngvd6ndEO+fHag4yVbRWqL3VYtDq9B/  
Ct6WaRBEa5rjQ1f7TyWutKdflbtVsgIWoaBFZxiFhoCNPgc548dPbOC9v3deOggvgmv8qDhTV2YI7/  
FGDkgPXxABhSA/x4QRzFeldbCmUnO8vuYUt0wTDOR9E3kjMYHl/X+pSiHX1ZIDCE/  
eVm7x95I4Iid6gtVLqx5lzcgcYEDMZ4qMHWrr/1a6FtywjbFA20ME1mY6t5Wu94xg9y19FZYy  
+RmhPo4DfAAiAFXJM/KhlrFLgEh7mDekP/YYHUDs7n/+tuLv27li/XhlozOt2UVZMXjwN1Y9fY0Xp9nJ3+UfR/  
nW5hvD5T1NnrLU0h2FN4YDYvOSUXzvvhbk1+4dglpe88F78ixhLKj1W04OAAWAj27oQzTvR+blQR/  
X06XOfXf5+d8gK41luU9llpFPXW+RWQIPoc23ehuM+otRoS/ef1vUnts4OMv7UVQJd  
+1PignXlkbvYfqBxqaMFN/7jCbza9RC+WQirOUkact3QQPNALiyyWF/Mdn8if6V  
+lVqw039AI8uU1I4Hw4cs7UesEtpKJPLGR6cdcgEE=
```

## Приклад розшифрованого повідомлення:



## Новий код частково покритий юніт-тестами:



**Висновок:** за результатами виконання цієї лабораторної роботи було ознайомлено з принципом роботи шифру RSA. Також, було проведене юніт-тестування коду.