Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №2

Black box testing

№ залікової книжки = 9301

9301 mod 6 = 1

Бібліотека: PasswordHashingUtils

Сирцевий код тестів

using System;

using Xunit;

using IIG.PasswordHashingUtils;

namespace PasswordHashingTests

{

public class Tests

{

[Fact]

public void test\_1\_1\_1()

{

string password = null;

string salt = "abc";

uint adlerMod32 = 12;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

Assert.Equal(expected, actual);

}

[Fact]

public void test\_1\_2\_1\_nullSalt()

{

string password = null;

string salt = null;

uint adlerMod32 = 12;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

Assert.Equal(expected, actual);

}

[Fact]

public void test\_1\_2\_1\_emptySalt()

{

string password = null;

string salt = "";

uint adlerMod32 = 12;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

Assert.Equal(expected, actual);

}

[Fact]

public void test\_1\_2\_2()

{

string password = null;

string salt = "abc";

uint adlerMod32 = 0;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

Assert.Equal(expected, actual);

}

[Fact]

public void test\_2\_1\_1()

{

string password = "password";

string salt = "abc";

uint adlerMod32 = 12;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

Assert.NotEqual(expected, actual);

}

[Fact]

public void test\_2\_2\_1\_nullsalt()

{

string password = "password";

string salt = null;

uint adlerMod32 = 12;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

Assert.NotEqual(expected, actual);

}

[Fact]

public void test\_2\_2\_1\_emptySalt()

{

string password = "password";

string salt = "";

uint adlerMod32 = 12;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

Assert.NotEqual(expected, actual);

}

[Fact]

public void test\_2\_2\_2\_emptySalt()

{

string password = "password";

string salt = "";

uint adlerMod32 = 0;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

Assert.NotEqual(expected, actual);

}

[Fact]

public void test\_2\_2\_2\_nullSalt()

{

string password = "password";

string salt = null;

uint adlerMod32 = 0;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

Assert.NotEqual(expected, actual);

}

[Fact]

public void test\_2\_1\_2()

{

string password = "password";

string salt = "salt";

uint adlerMod32 = 78;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

Assert.NotEqual(expected, actual);

}

[Fact]

public void test\_1\_1\_2()

{

string password = null;

string salt = "salt";

uint adlerMod32 = 0;

string expected = null;

string actual = PasswordHasher.GetHash(password, salt, adlerMod32);

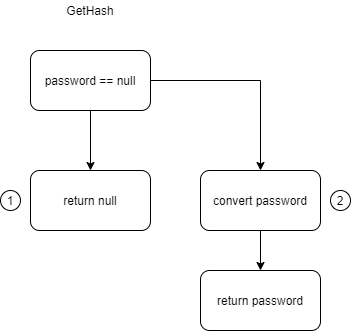
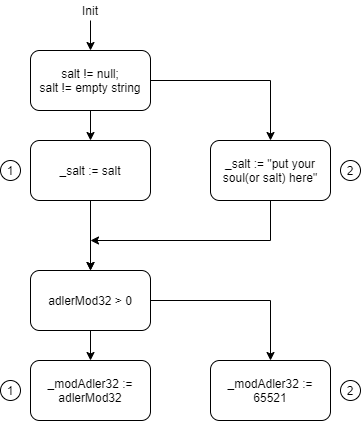
Assert.Equal(expected, actual);

}

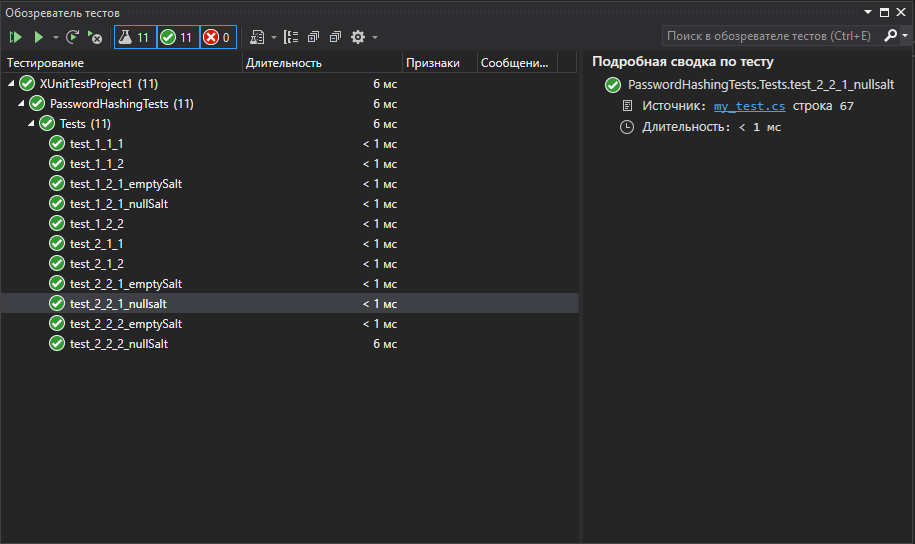
}

}

Діаграми методів:



Результат тестів:



У white box використовував тестування потоку виконання. За допомогою діаграм можна перевірити максимально можливу кількість шляхів виконання пз(це можливо завдяки тому що в white box у нас є доступ до коду і ми розуміємо як він працює)