

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**Український державний університет науки і технологій**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №1  
з дисципліни «Безпека програм та даних»**

**на тему:**

**«**Захист програмного забезпечення від несанкціонованого використання.»

Виконав:

студент гр. ПЗ2011

Проценко Р.О.

Прийняв:

доц. Жеваго О. О.

Дніпро, 2023

**Тема.** Захист програмного забезпечення від несанкціонованого використання.

**Мета.** Ознайомитись з основними методами захисту програмного забезпечення від несанкціонованого використання.

**Завдання**

1. Розробити програмний продукт, що виконує мінімум 5 функцій.

2. Реалізувати модель розповсюдження програмного забезпечення згідно варіанту.

3. Реалізувати шифрування ключа, який розблоковує функції системи.

**Індивідуальне завдання**

Trialwаre. Блокування виконання програми через 30 днів після першого використання.

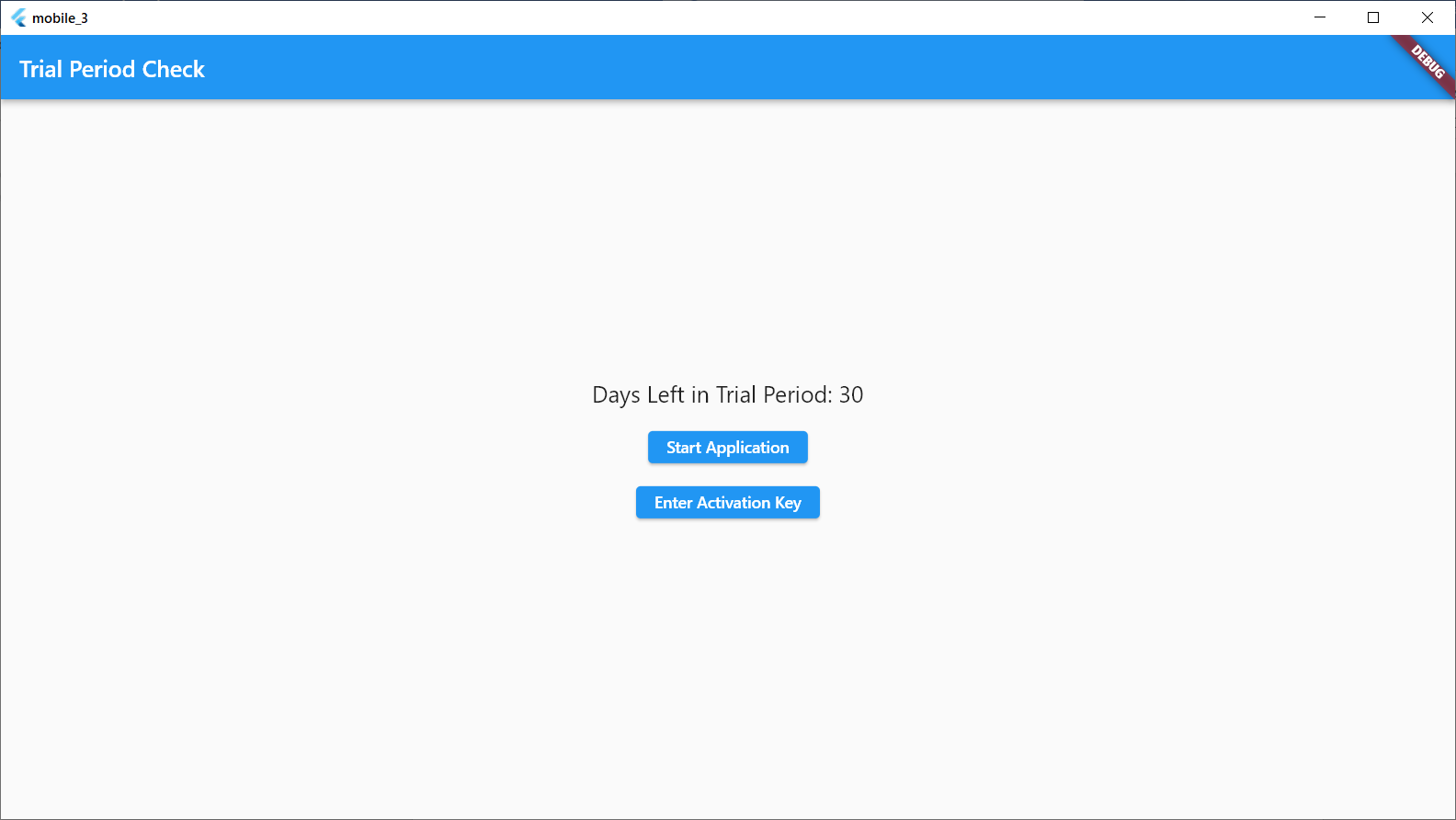
**Текст програми**

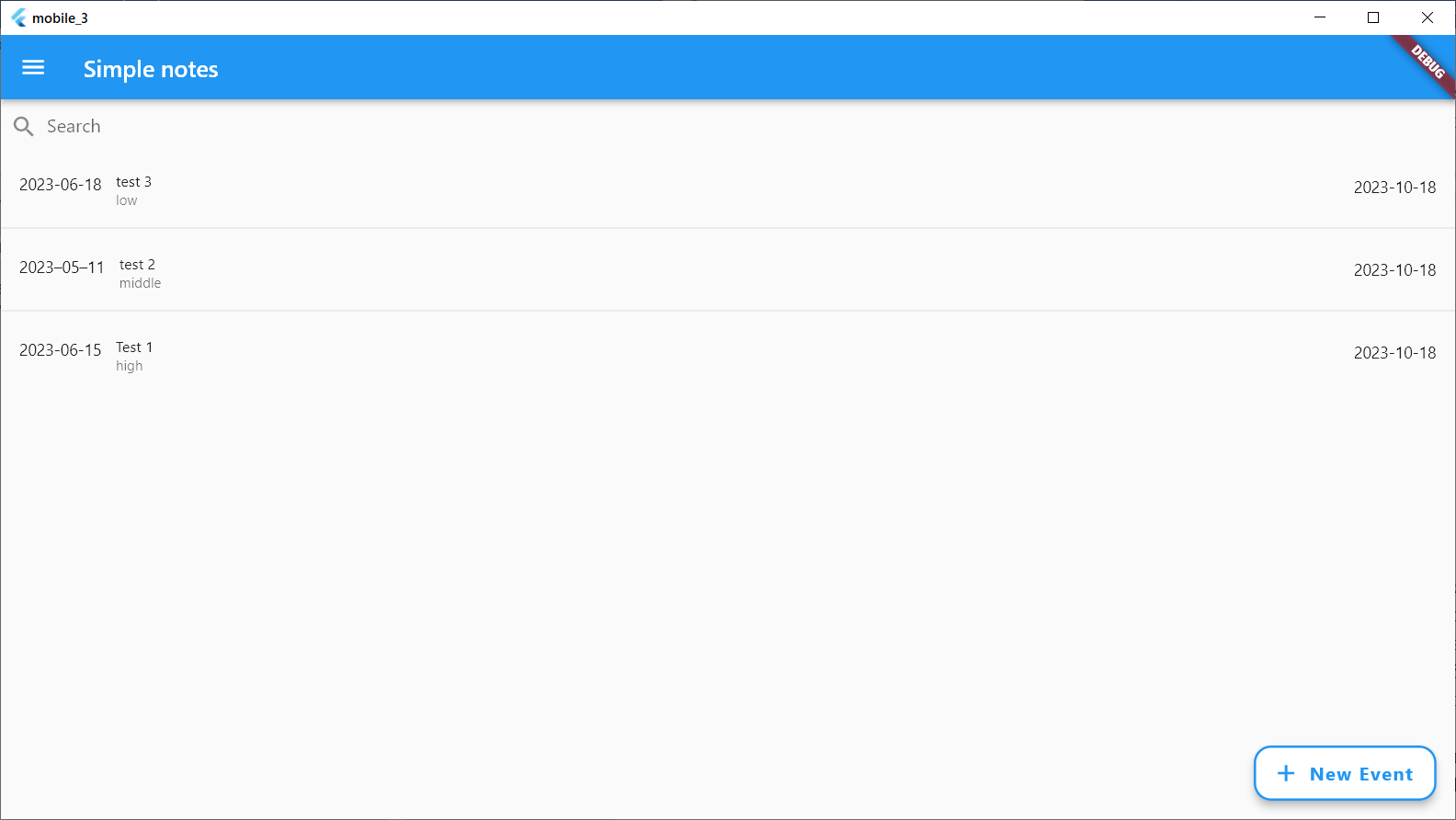
**class** MyApp **extends** StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 **return** MaterialApp(  
 home: TrialPeriodChecker(),  
 );  
 }  
}  
  
**class** TrialPeriodChecker **extends** StatefulWidget {  
 @override  
 \_TrialPeriodCheckerState createState() => \_TrialPeriodCheckerState();  
}  
  
**class** \_TrialPeriodCheckerState **extends** State<TrialPeriodChecker> {  
 int **daysLeft** = 30;  
  
 @override  
 **void** initState() {  
 **super**.initState();  
 checkTrialPeriod();  
 }  
  
 Future<**void**> checkTrialPeriod() **async** {  
 **final** prefs = **await** SharedPreferences.*getInstance*();  
  
 *// Створення екземпляру Encrypter з використанням AES (CBC)* String key = **"secret\_key\_roman"**;  
 **final** iv = encrypt.IV.fromUtf8(**"secret\_key\_roman"**);  
 print(iv);  
 **final** encrypter = encrypt.Encrypter(encrypt.AES(encrypt.Key.fromUtf8(key), mode: encrypt.AESMode.**cbc**));  
  
 **final** firstLaunchDate = prefs.getString(**'firstLaunchDate'**);  
 **final** \_firstLaunchDate = firstLaunchDate != **null** ? int.*parse*(encrypter.decrypt(encrypt.Encrypted.fromBase64(firstLaunchDate), iv: iv)) : **null**;  
 **final** activate = prefs.getString(**"activateLicence"**);  
 **final** \_activate = activate != **null** ? bool.*parse*(encrypter.decrypt(encrypt.Encrypted.fromBase64(activate), iv: iv)) : **false**;  
  
 **var** currentDate=-1;  
 **final** response = **await** http.get(Uri.*parse*(**'http://worldclockapi.com/api/json/utc/now'**));  
  
 **if** (response.**statusCode** == 200) {  
 **final** Map<String, **dynamic**> data = json.decode(response.**body**);  
 **final** String currentDateTime = data[**'currentDateTime'**];  
 DateFormat format = DateFormat(**"yyyy-MM-dd'T'HH:mm'Z'"**);  
 DateTime dateTime = format.parse(currentDateTime);  
 currentDate = dateTime.**millisecondsSinceEpoch**;  
 } **else** {  
 print(**'Failed to fetch time data'**);  
 }  
  
 **if** (activate != **null** && \_activate) {  
 runApp(MyAppStart());  
 }  
  
 **if** (\_firstLaunchDate == **null**) {  
 **final** \_currentDate = encrypter.encrypt(currentDate.toString(), iv: iv);  
 prefs.setString(**'firstLaunchDate'**, \_currentDate.**base64**);  
 setState(() {  
 **daysLeft** = 30;  
 });  
 } **else** {  
 **final** elapsedTime = currentDate - \_firstLaunchDate;  
 **final** remainingDays = 30 - (elapsedTime ~/ (1000 \* 3600\*24));  
 setState(() {  
 **daysLeft** = remainingDays;  
 });  
 }  
 }  
  
 **void** startApp() {  
 runApp(MyAppStart());  
 }  
  
 **void** showInputDialog() {  
 TextEditingController keyController = TextEditingController();  
 showDialog(  
 context: **context**,  
 builder: (BuildContext context) {  
 **return** AlertDialog(  
 title: Text(**'Enter Product Activation Key'**),  
 content: TextField(  
 controller: keyController,  
 decoration: InputDecoration(hintText: **'Key'**),  
 ),  
 actions: <Widget>[  
 TextButton(  
 onPressed: () {  
 Navigator.*of*(context).pop();  
 },  
 child: Text(**'Cancel'**),  
 ),  
 TextButton(  
 onPressed: () **async** {  
 **final** iv = encrypt.IV.fromUtf8(**"secret\_key\_roman"**);  
 String key = **"secret\_key\_roman"**;  
 **final** encrypter = encrypt.Encrypter(encrypt.AES(encrypt.Key.fromUtf8(key),  
 mode: encrypt.AESMode.**cbc**));  
  
 String enteredKey = keyController.**text**.toString();  
 **final** encryptedKey = encrypter.encrypt(enteredKey, iv: iv);  
  
 **final** prefs = **await** SharedPreferences.*getInstance*();  
  
 **final** \_\_key = prefs.getString(**"encryptedKey"**);  
  
 **if** (encryptedKey.**base64** == \_\_key) {  
 Navigator.*of*(context).pop();  
 startApp();  
  
  
 **final** \_activated = encrypter.encrypt(**"true"**, iv: iv);  
 prefs.setString(**'activateLicence'**, \_activated.**base64**);  
 } **else** {  
 ScaffoldMessenger.*of*(context).showSnackBar(  
 SnackBar(  
 content: Text(**'Invalid Key!'**),  
 ),  
 );  
 }  
 },  
 child: Text(**'OK'**),  
 ),  
 ],  
 );  
 },  
 );  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 **return** Scaffold(  
 appBar: AppBar(  
 title: Text(**'Trial Period Check'**),  
 ),  
 body: Center(  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.**center**,  
 children: <Widget>[  
 Padding(  
 padding: EdgeInsets.only(bottom: 20.0), *// Add padding to the bottom* child: Text(  
 (**daysLeft** > 0) ? **'Days Left in Trial Period:** $**daysLeft'** : **'Trial Period Left!'**,  
 style: TextStyle(fontSize: 20),  
 ),  
 ),  
 **if** (**daysLeft** > 0)  
 Padding(  
 padding: EdgeInsets.only(bottom: 20.0), *// Add padding to the bottom* child: ElevatedButton(  
 onPressed: startApp,  
 child: Text(**'Start Application'**),  
 ),  
 ),  
 Padding(  
 padding: EdgeInsets.only(bottom: 20.0), *// Add padding to the bottom* child: ElevatedButton(  
 onPressed: showInputDialog,  
 child: Text(**'Enter Activation Key'**),  
 ),  
 ),  
  
 ],  
 ),  
 ),  
 );  
 }  
}  
  
**void** main() {  
 runApp(MyApp());  
}

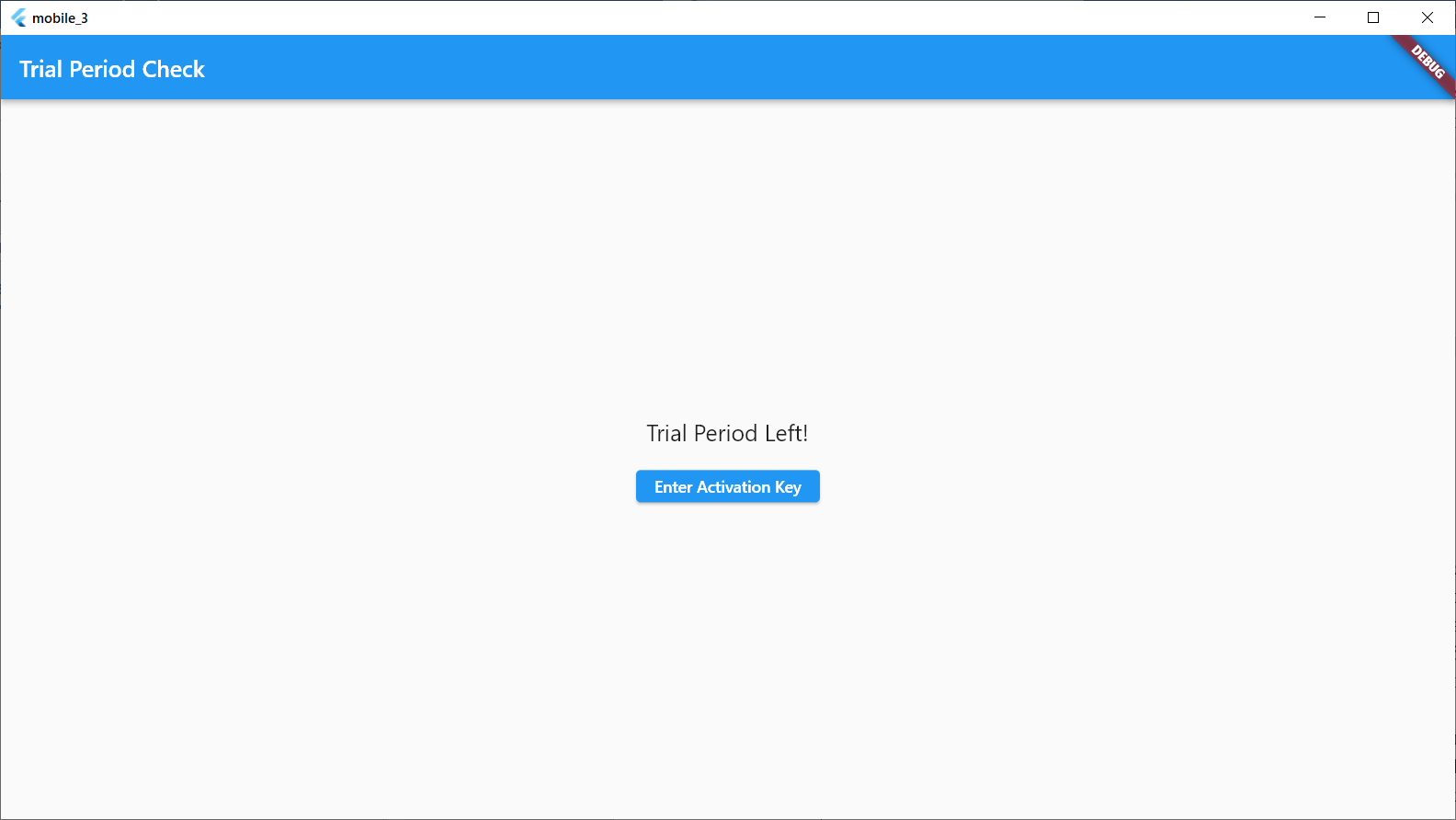
**Опис як відбувається захист програмного забезпечення від несанкціонованого використання**

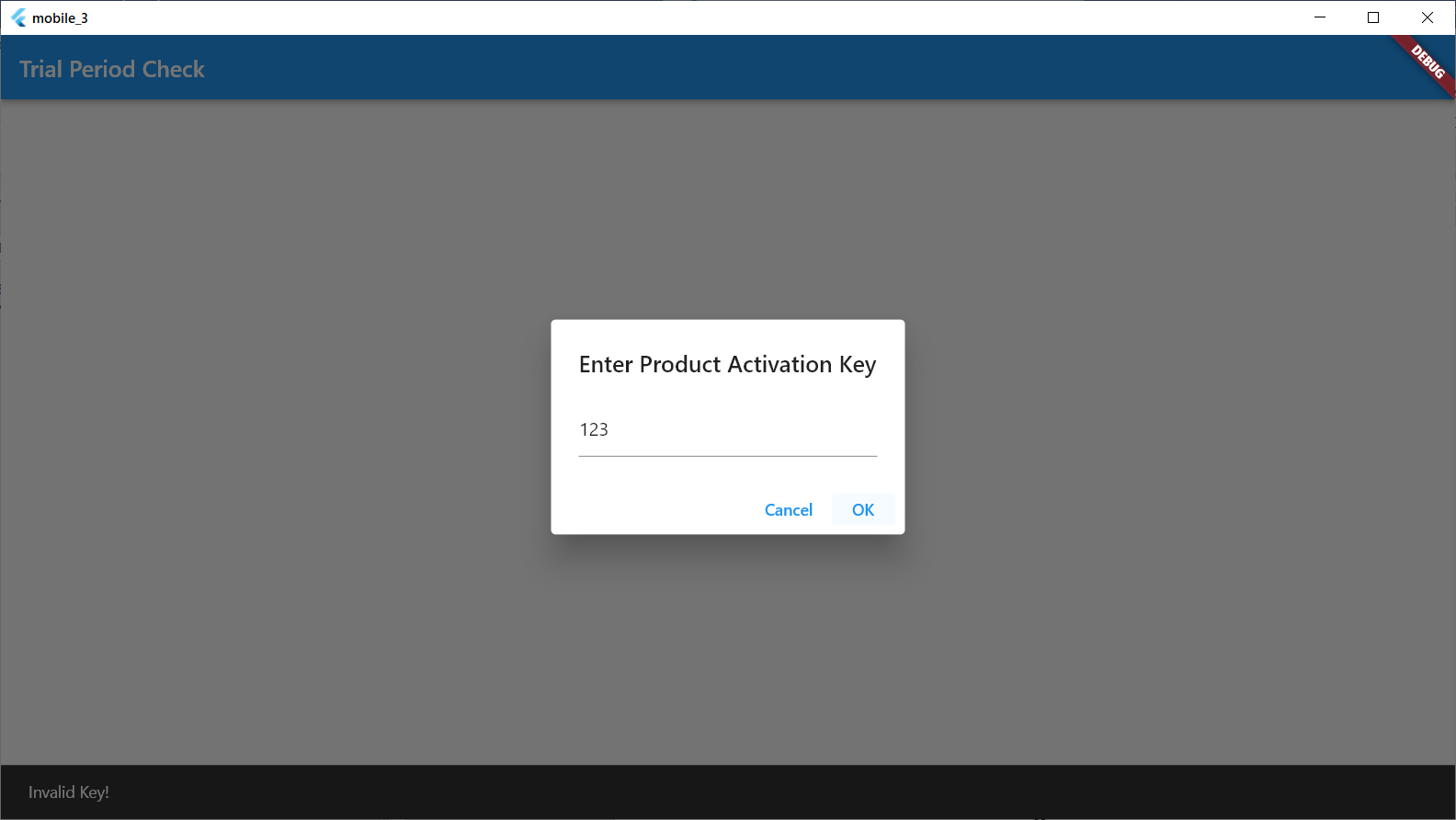
Відповідно до індивідуального завдання необхідно реалізувати trialware, а саме: блокування виконання програми через 30 днів після першого використання. Реалізовано це наступним чином: при запуску програми з'являється нагадування, скільки днів тріального режиму у користувача залишилося. Після лише можливий перехід до програмного продукту.

У разі, якщо тріальний період у 30 днів скінчився, кнопка перейти до програми заміниться на кнопку ввести ключ активації. При натисненні на неї з’являється форма для введення ключа. У разі, якщо ключ був введений вірно, програма продовжить своє виконання, і в майбутньому, при запусках не виводитиме ні вікна про тріальний режим, ні вікна введення ключа продукта. У разі якщо ключ було введено не вірно, з’явиться відповідне повідомлення, і користувач матиме можливість знову спробувати ввести ключ, або закрити програмний продукт.









**Опис як відбувається шифрування ключа, і як зберігається інформація про використання програми**

Шифрування ключа у програмі відбувається за допомогою бібліотеки ***encrypt.*** Це бібліотека, реалізована на мові програмування дарт, та створенна для шифрування ключів.

Нижче наведено приклад кодування:

* ключ активації в незакодованому вигляді: super\_key;
* ключ активації в закодованому вигляді: tHh1EY6W3TTP3+HWnq+Rvg==

Інформація про використання програми зберігається за допомогою бібліотеки shared\_preferences в зашифрованому вигляді ща допомогою бібліотеки encrypt, зберігання інформації виконується у папку AppData, а коли треба, за допомогою бібліотеки витягується та використовується. Приклад зашифрованної інформації:



Окрім цього дані про поточний час, необхідний для розрахунку часу беруться з апі за адресою: **'http://worldclockapi.com/api/json/utc/now'.**

**Висновки**

Сьогодні існують три основні моделі розповсюдження програмного забезпечення: безкоштовне, умовно безкоштовне і комерційне. Безкоштовне ПЗ не вимагає оплати і часто розробляється ентузіастами для користування спільнотою. Умовно безкоштовне ПЗ надає можливість оцінити продукт перед покупкою, з обмеженнями в часі або функціональності. Комерційне ПЗ продається тільки за гроші перед отриманням доступу.

Умовно безкоштовні продукти можуть використовувати різні методи обмеження, такі як демо-версії, пробні періоди і нагадування про придбання. Захист комерційного ПЗ включає перевірку ліцензійних ключів та може використовувати шифрування ключів для підвищення безпеки.