ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ СЕМЕНА КУЗНЕЦЯ

Лабораторна робота №5

з курсу «Безпека банківських систем»

РОЗРОБКА СИСТЕМИ «БАНКОМАТИК»

Харків 2023

Мета: ознайомитися з основами програмного забезпечення та логіки роботи банкоматів, розробити програмне забезпечення, що моделює роботу пристрою.

1 ТЕОРЕТИЧНІ ВІДОМОСТІ

1.1 Будова банкомату

Будь-який банкомат по суті є комп'ютером з підключеною периферією, менеджером обладнання і власне банківським застосунком, що управляє всім цим набором. Усі рішення щодо видачі грошей приймає сервер. Банкомат лише збирає інформацію від клієнта та передає її на сервер.

Мінімальний набір «заліза» банкомату включає:

– картридер для читання картки клієнта;

– пін-пад, для введення пін-коду та іншої інформації, як, наприклад, суми платежу/зняття;

– функціональні клавіші з боків (4+4) є доповненням, що підключається до пін-паду. У деяких сучасних банкоматах їх замінили на тач-скрин;

– диспенсер для видачі грошей;

– різні датчики, підсвічування.

1.2 Як відбувається керування

Для того, щоб виробники не витрачали час на написання драйверів, які потім нікому крім них не потрібні, а розробники софту не страждали від різноманітності рішень щодо управління тією чи іншою «зялізякою», було вирішено всі ці завдання уніфікувати.

Було створено стандарт CEN/XFS або просто XFS, що розшифровується як eXtension For Financial Services.

Стандарт описує клієнт-серверну архітектуру, що складається з менеджера обладнання та сервісних провайдерів (читай драйверів пристроїв),

якими він керує. У термінології стандарту «сервісний провайдер» – це бібліотека, що надає певний набір функцій для отримання інформації про пристрій та керування ним. Зазвичай це динамічна бібліотека, що містить певний набір стандартних функцій (*Open*, *Close*, *GetInfo*, *Execute*), кожна з яких має ряд специфічних для конкретного пристрою аргументів.

Вся взаємодія з обладнанням відбувається через API менеджера XFS. Наприклад, параметр *Command* функції *Execute* може мати значення для диспенсера купюр:

WFS_CMD_CDM_DISPENSE (забір купюр з касет)

WFS_CMD_CDM_PRESENT (видача купюр клієнту)

Для картридера:

WFS_CMD_IDC_RETAIN_CARD (захоплення картки),

WFS_CMD_IDC_READ_TRACK (читання доріжок)

Існує кілька реалізацій XFS-менеджерів (у тому числі з відкритим вихідним кодом), написаних на C++ та бібліотеки сервісних провайдерів, написані під один менеджер, які так само повинні підходити до всіх інших. Але за фактом іноді бібліотека, написана конкретним вендором під конкретний XFS менеджер, працює лише з цим менеджером.

Також існує Java XFS зі своїми бібліотеками, які не сумісні з класичними менеджерами.

1.3 Банківський застосунок

Банківська програма – це те, що ви бачите на екрані, коли підходите до пристрою. Вона призначена для збору даних від користувача, надсилання цих даних на хост (сервер) та виконання відповіді від хоста. Як і у випадку із «залізом» (XFS) є галузеві протоколи (NDC/DDC), якими застосунок спілкується з хостом, завантажує конфігурацію та інтерпретує її.

Будь-який великий виробник банкоматів (Wincor, NCR, Diebold) має власну реалізацію як XFS, і банківського застосунку.

Однак на ринку є альтернативний софт, що відповідає всім стандартам і не прив'язаний до конкретного вендору.

Далі описуватиметься банкомат з прикладу NDC як найпоширенішого в Україні протоколу, але трохи менш популярний DDC має схожий принцип роботи.

1.3.1 Принцип роботи

У кожний момент часу банкомат знаходиться в одному з режимів роботи:

– Power Up – завантаження;

– Offline – немає зв'язку з сервером, виконується конект;

– Supervisor – працює інкасатор або сервіс-інженер;

– Out of service – банкомат не працює, або несправний, чи скінчилися гроші, чи просто хтось у банку перевів його в цей режим;

– In service – основний режим роботи, знайомий усім тим, хто має банківські картки.

У режимі In service банкомат знаходиться в одному зі станів (стейтів), з номером від 001 до 999, та 25 символьним рядком-описом.

Перший символ цього рядка – тип стейту (позначаються літерами A...Z а також a...z і деякими символами (,'.?)), він визначає сукупність. Інші 24 символи – це 8 десяткових 3-значних чисел, кожне з яких є певним налаштуванням стейту (номер екрана для показу, умови переходу на стейт, список дій). Стейтів одного типу може бути будь-яка кількість.

1.3.2 Режим In service

При старті режиму обслуговування банкомат автоматично починає виконувати стейт 000. Зазвичай, це стейт A (Card read state). У цьому стейті банкомат відображає екран із запрошенням вставити картку та переводить картридер у режим прийому. Також стейт відповідає за читання карти та розгалуження залежно від результатів цієї операції.

Нижче наведено приклад конфігурації типового стейту A:

000 *A0010010110080040020011104*

000 – номер стейту;

A – тип стейту (Card read state);

001 – номер екрану (Screen number);

001 – номер стейту, на який виконується перехід у разі успішного читання картки;

011 – номер стейту, на який виконується перехід у разі помилок читання картки;

008 – умова читання 1;

004 – умова читання 2;

002 – умова читання 3;

001 – умова повернення картки (відразу після читання або після завершення операції);

104 – стейт переходу, якщо картка невідома банку.

Пройдемося за параметрами докладніше:

Тип стейту – тут все зрозуміло: визначивши тип стейту, застосунок знає як інтерпретувати подальші параметри.

Номер екрану – це посилання на рядок з текстовим описом екрана, що відображається під час роботи даного стейту.

Не кожен стейт має екран.

Екран може мати номер від 000 до 999. Екрани, що відрізняються від 100, зазвичай резервують під різні мови. Таким чином, екран 010 і екран 210 це швидше за все різномовні версії одного екрана. Екрани розглянемо трохи згодом.

Номер стейта переходу у разі успішного читання карти – те, який стейт програма почне виконувати у випадку, якщо картку розпізнано та дані прочитані успішно.

Окрім стейтів та екранів у банкоматі є ще один важливий конфігураційний параметр – financial institution table. Таблиця фінансових інститутів містить дані про те, які карти належать якому банку, як правильно

парсити дані прочитані з доріжок картки, і що робити в залежності від цих даних далі. Наприклад, якщо картка локальна, то можна виконати один сценарій, якщо карта стороннього банку то потрібно заборонити гілку сценарію з мобільними платежами та перевіркою балансу.

Номер стейту переходу у разі проблем із читанням карти – якщо картку не вдалося прочитати за жодною із запропонованих умов – переходимо на стейт, зазначений у цьому параметрі. Як правило, це стейт J (Close state) на якому віддаємо карту, показуємо екран із пропозицією забрати її та активуємо таймер після закінчення терміну якого буде запущено механізм утримання картки. Стейт J також є останнім стейтом у разі успішної транзакції.

Умови читання карти (3 параметри поспіль) – це бітові маски, що позначають номери треків, які потрібно прочитати, та взаємодія з чіпом у разі його наявності.

Наприклад, Read Chip, Read Track 2 and Track 1, Read Track 1. Якщо хоч одна з умов спрацьовує, інші умови не виконуються і карта вважається прочитаною. Якщо жодна з умов не виконується, картка вважається непрочитаною.

Умова повернення картки – банкомат може повернути картки відразу після прочитання, а може зробити це наприкінці після завершення всіх операцій.

Інші стейти влаштовані таким чином:

– є стейти для читання суми з клавіатури та поміщення у спеціальний внутрішній буфер;

– є стейти для читання пін-коду пін-падом та отримання потім пін-блоку у спеціальний буфер;

– є стейти для перевірки введених даних (наприклад, якщо введена сума менша за мінімальну суму, то йде перенаправлення на стейт з повідомленням про помилку);

– є стейти для вибору за допомогою бічних клавіш (так званих FDK) та розміщення символів цих клавіш (ABCD FGHI) у спеціальний 8-байтний буфур;

– є стейти для обнулення та встановлення буферів.

Переходячи по всіх цих стейтах, застосунок рано чи пізно доходить до стейту взаємодії з хостом – стейту I (Transaction Request State). На цьому стейті формується запит даних, зібраних на минулих стейтах і відправляється на сервер. Запит являє собою ID банкомату (Logical Unit Number), дані з доріжок картки, дані про попередні транзакції, дані з буферів суми, пін-блоки, натискання функціональних клавіш (FDK буфер). Дані поділяються символом роздільника. Серверний застосунок отримує запит і аналізує буфер FDK – саме за вмістом цього буфера хост розуміє, чого хоче банкомат. Після чого, залежно від прийнятого рішення, відсилає відповідь, в якій містяться:

– ідентифікатор дії, яку потрібно виконати;

– номер екрану, який потрібно при цій дії показувати;

– вміст чека, якщо чек потрібно надрукувати;

– стейт на який потрібно перейти по завершенню дії.

У спеціальному буфері передається кількість купюр, які потрібно видати з кожної касети (якщо це операція зняття готівки). Саме кількість купюр, тому що банкомат не знає номінали грошей, що видаються для нього, це просто папірці в касетах.

Після завершення необхідних дій програма надсилає підтвердження на хост і переходить на вказаний стейт. Як правило, це вже відомий нам стейт J. У разі будь-якого збою застосунок шле повідомлення про збій на хост і чекає нового Transaction Reply з переходом на новий стейт.

1.4 Екран банкомату

Екран банкомату є полем 32x16 клітин. Екран може містити як графічну інформацію, так і текстову, яка позиціонується щодо клітин. Шрифти можуть бути подвійної висоти.

Опис екрана являє собою рядок з текстом, що перемежовується символами управління, такими як очищення екрана, позиціонування курсору, розмір шрифту. У більшості сучасних банків текст використовується тільки при введенні сум, а в інших випадках екран це просто цілісна картинка. Проте трапляються й повністю текстові екрани.

Приклад екрана, що відображає картинку з таблиці картинок (\0c\1bP2018\1b\5c)

Саме такі екрани посилаються параметри стейта.

Сукупність стейтів, екранів, FIT, таймерів називається сценарієм банкомату. Кожен сценарій має власний номер. Після завантаження банкомату і підключення його до мережі, він шле на хост повідомлення, в якому повідомляє свій ID і номер конфігурації. Якщо конфігурацію слід оновити – хост переводить банкомат у режим Out Of Service і починає завантажувати необхідні параметри нової конфігурації. Останнім параметром є номер конфігурації. Так само відбувається завантаження ключів для шифрування пін-блоку, для маскування, і майстер-ключів.

За таким принципом організовано роботу банкомату.

2 ХІД РОБОТИ

Завданням даної лабораторної роботи є розробка програмного забезпечення, моделюючого роботу банкомата. Дане програмне забезпечення може бути корисним елементом навчання власників банківських карток для їх правильного використання.

Розроблене програмне забезпечення дозволить не залишитися на узбіччі життя тим власникам банківських карток, хто через вік і специфіку роботи не є досвідченим користувачем сучасних обчислювальних систем.

2.1 Виділення функціональних елементів

Програмне забезпечення, що створюється, має складатися не менше ніж із двох файлів з кодом (*.*cpp* при умові використання C++), об'єднаних у проект. Крім того, повинен бути включений хоча б один заголовний файл.

Відповідно до цього виділимо два файли: *Screen.cpp* і *ProjectBankomat.cpp*. У файлі *Screen.cpp* будуть описані функції заставки. У файлі *ProjectBankomat.cpp* будуть описані функції моделювання роботи банкомату. Кожен з них має свій заголовний файл: *Bankomat.h* та *Screen.h* відповідно. Ці файли об'єднані у проект, параметри якого описані у файлі *ProjectBankomat.bpr.*

Виділимо основні функціональні частини роботи програмного забезпечення, що розробляється:

– ініціалізація. Ця функціональна частина відповідає за читання параметрів банківських карток із файлу;

– вставляння картки. Ця функціональна частина відповідає за вибір банківської картки, «ув'язування» її з банкоматом та авторизацію (введення ПІН-коду);

– основне меню. Ця функціональна частина відповідає за вибір операції з банківською карткою (видача готівки, перегляд балансу, платежі);

– видача готівки. Ця функціональна частина відповідає за видачу готівки власнику банківської картки відповідно до запиту;

– перегляд балансу. Ця функціональна частина відповідає за інформування власника банківської картки про наявність коштів на балансі;

– платежі. Ця функціональна частина відповідає за виконання платежів за телефон. При цьому користувач повинен мати можливість вибору мобільного оператора.

Ці функціональні частини виконують певні операції з банківськими картками. Послідовність виклику даних функціональних частин визначатиметься алгоритмом роботи програми.

2.2 Розробка алгоритму роботи програми

Вище за текстом наше програмне забезпечення було поділено на функціональні частини. Усі вони виконують певні операції з банківською карткою та взаємодіють між собою відповідно до певного алгоритму.

Першою функціональною частиною, якій передається керування під час запуску програмного забезпечення, є модуль ініціалізації. Після зчитування параметрів карток з файлів налаштувань управління передається функціональній частині, що відповідає за вставку картки. Вибравши одну з двох карток, необхідно провести «авторизацію» – ввести ПІН-код до картки. У разі правильного введення управління буде передано в основне меню, де користувач повинен вибрати вид операції над банківською карткою: видача готівки, перегляд балансу, платежі. Також тут можливий вибір завершення роботи з банківською карткою – вихід із «поверненням» банківської картки.

При виборі операції видачі готівки управління програмою від основного меню передається функціональній частині видачі готівки. Ця функціональна частина відповідає за запит суми для видачі, перевірку наявності цієї суми на рахунку власника банківської картки, визначення доступності коштів у банкоматі та безпосередньо за визначення кількості банкнот та їх номіналу.

Після виконання «видачі» грошей управління передається назад в основне меню.

При виборі операції визначення балансу управління програмою від основного меню передається функціональній частині визначення балансу. Ця функціональна частина відповідає за запит суми коштів на рахунку власника банківської картки та виведення її на екран. Після виконання виведення суми грошей управління передається назад в основне меню.

При виборі операції виконання платежів керування від основного меню передається функціональній частині виконання платежів. Ця функціональна частина відповідає за визначення оператора мобільного зв'язку, запит номера телефону та суми платежу. За наявності на балансі зазначеної суми коштів провадиться оплата. Після виконання платежу управління передається назад в основне меню.

2.3 Опис готового застосунку

Програму, роботу якої буде описано в даному розділі, було створено на мові програмування C++. *Під час розробки ви можете використовувати будь-яку мову програмування на ваш вибір.*

Під час запуску програмного забезпечення на екрані з'являється інформаційна заставка (рис. 2.1).

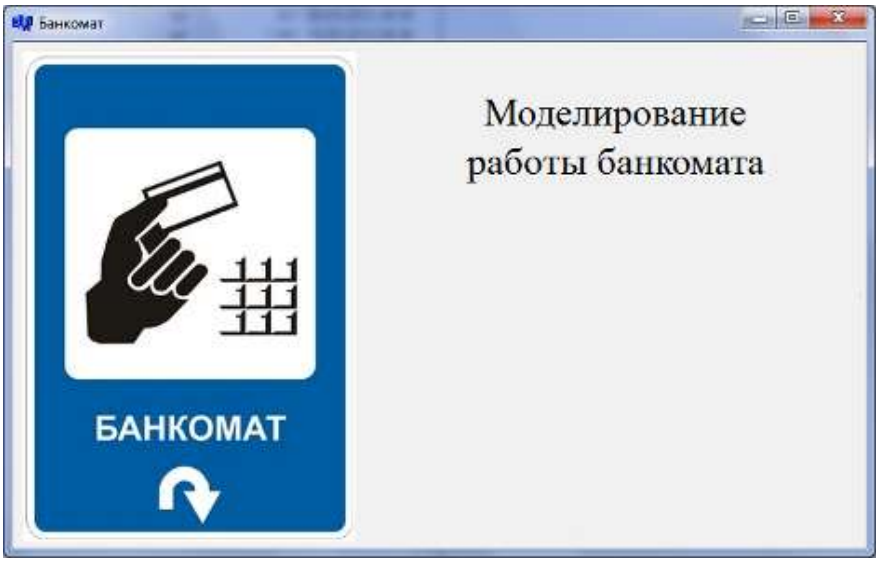

Рисунок 2.1 – Інформаційна заставка

Після показу заставки управління передається в основне вікно. Вид основного вікна представлений рис. 2.2.

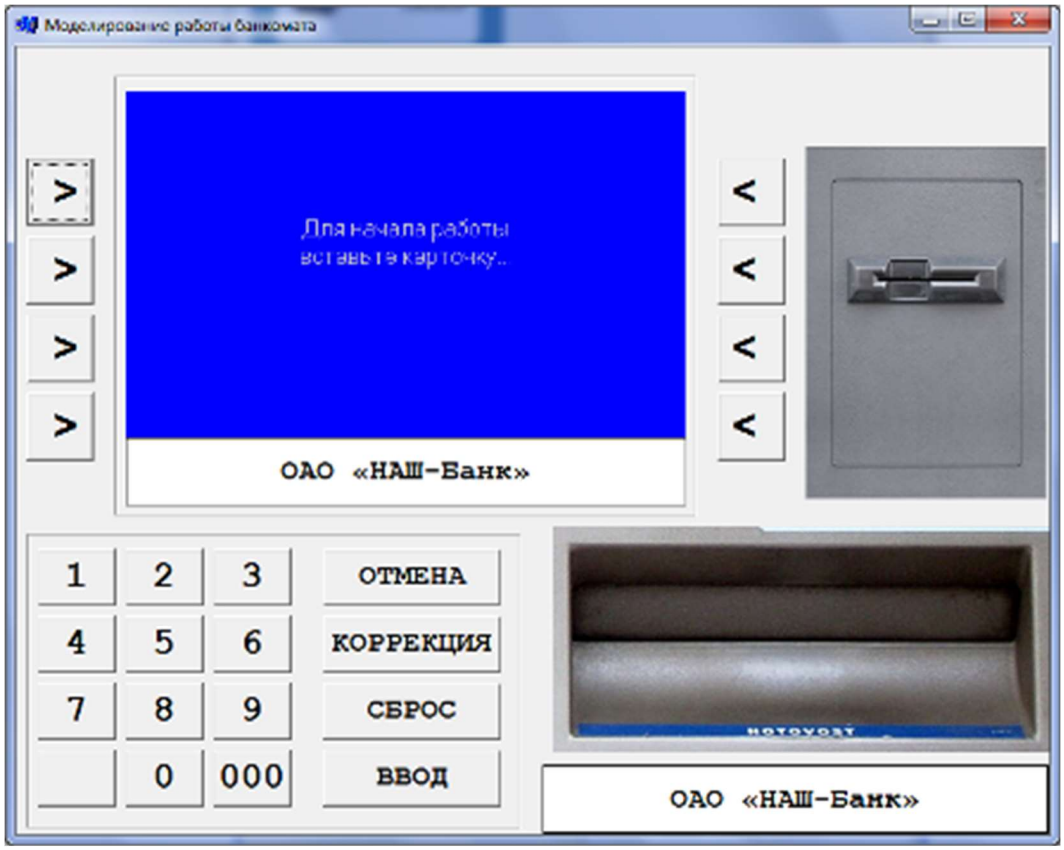

Рисунок 2.2 – Основне вікно

У програмі реалізовано можливість «вставки» до банкомату однієї з двох карток. Вибір картки реалізовано за допомогою контекстного меню. Процес вибору картки представлений рис. 2.3.

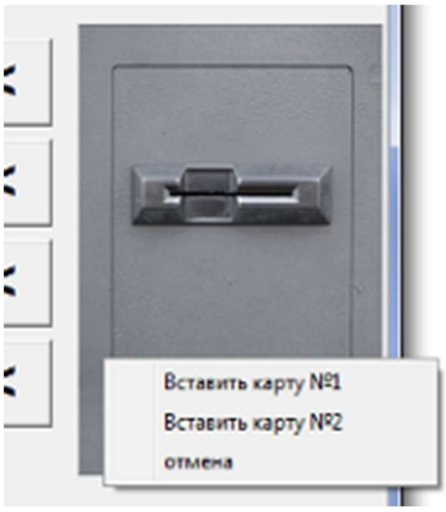

Рисунок 2.3 – Процес вибору картки

Після вставки картки починається процес авторизації (запит ПІН-коду). Для картки №1 ПІН-код встановлено 1234, для картки №2 – 4321. Вікно для введення ПІН коду представлено рис. 2.4.

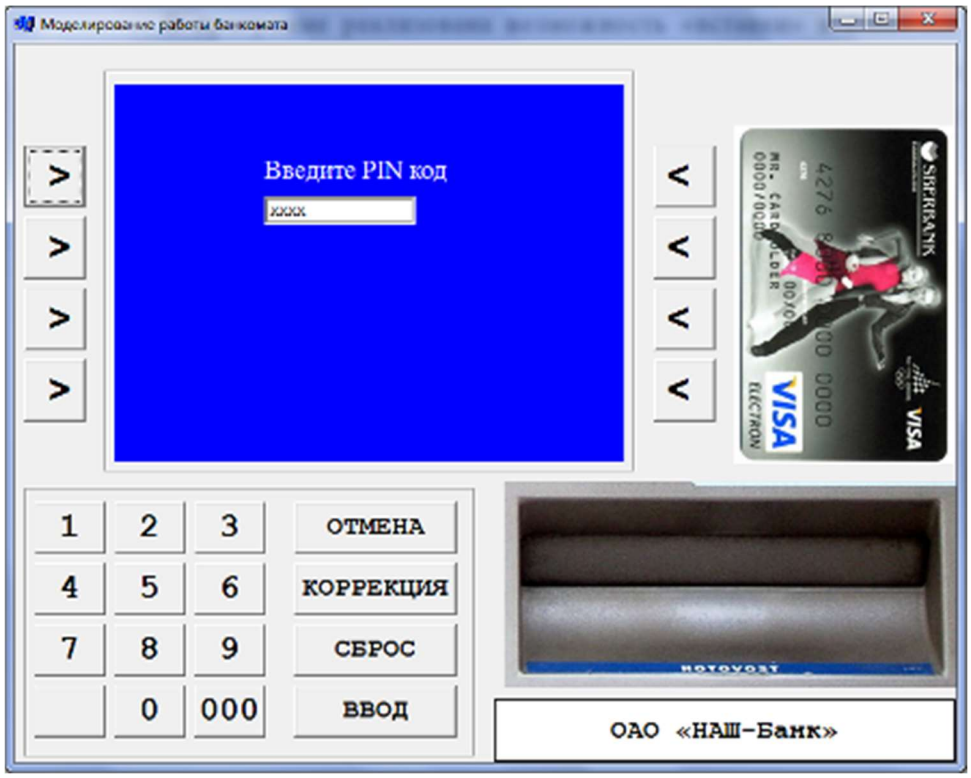

Рисунок 2.4 – Вікно для введення ПІН-коду

Після введення PIN-коду картки на екрані з'являється основне меню. Основне меню представлене рис. 2.5.

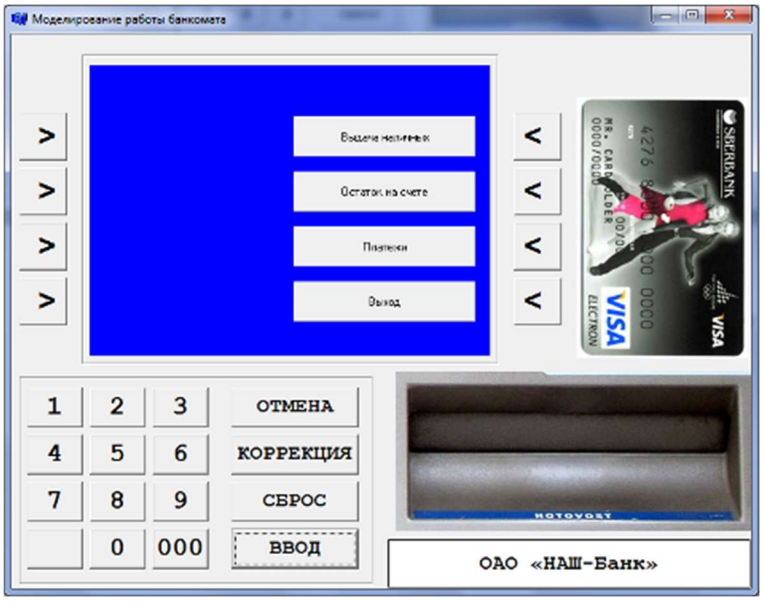

Рисунок 2.5 – Основне меню

При виборі операції видачі готівки здійснюється запит суми (рис. 2.6). У разі коли запитана сума перевищує баланс коштів на банківській картці – користувач побачить відповідне повідомлення про помилку (рис. 2.7). В іншому випадку буде виконано перевірку на наявність коштів у банкоматі та перевірку на правильність вводу суми. Якщо сума не є коректною – користувач побачить відповідне повідомлення про помилку (рис. 2.7).

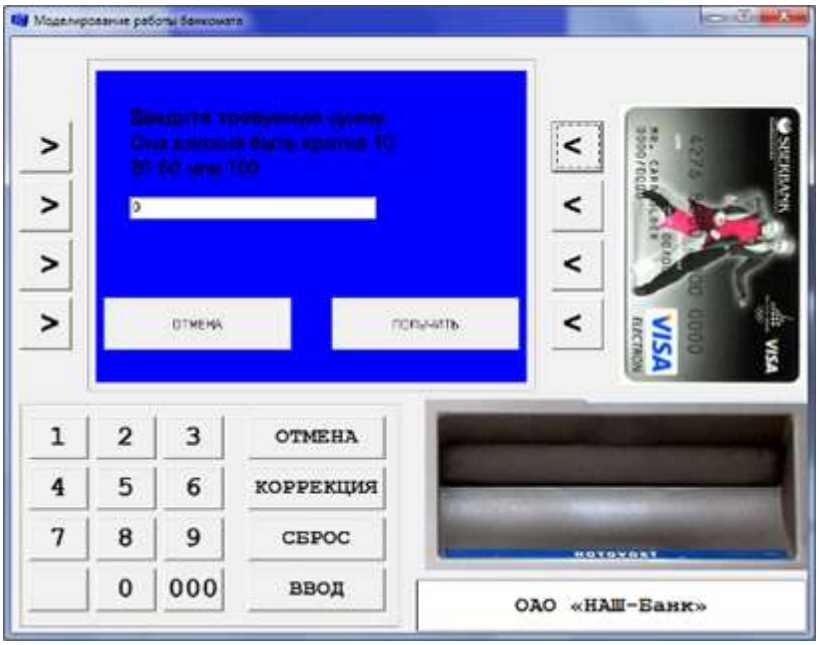

Рисунок 2.6 – Вікно запиту суми

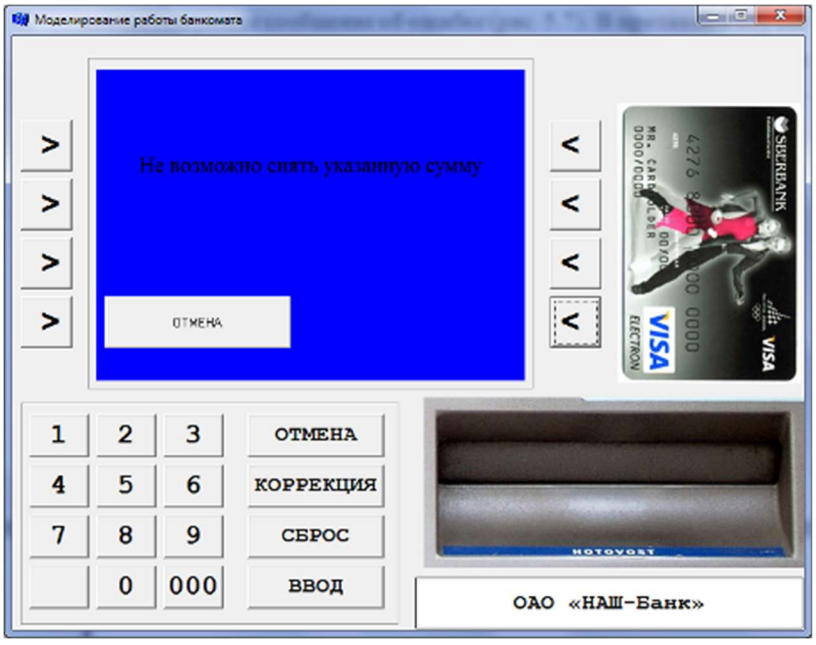

Рисунок 2.7 – Вікно повідомлення про помилку

У разі правильного введення даних користувачу будуть видані гроші (рис. 2.8).

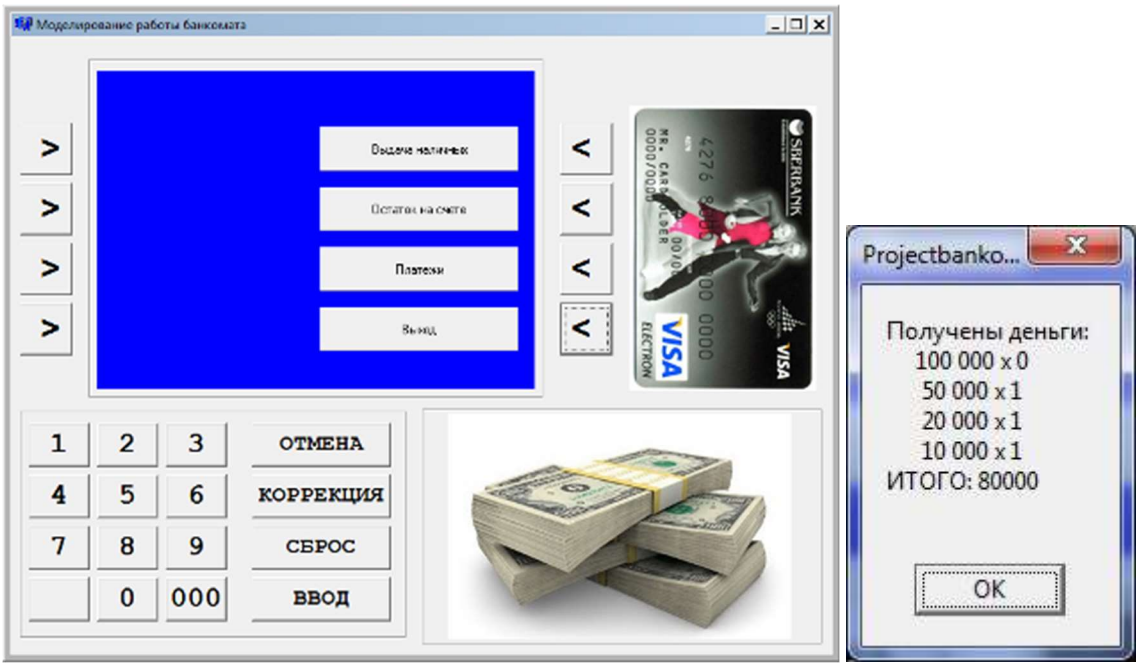

Рисунок 2.8 – «Видача» коштів

Після видачі грошей програма повертається до основного меню. При виборі операції перевірки залишку проводиться запит наявності коштів на рахунку та виведення даних на екран (рис. 2.9).

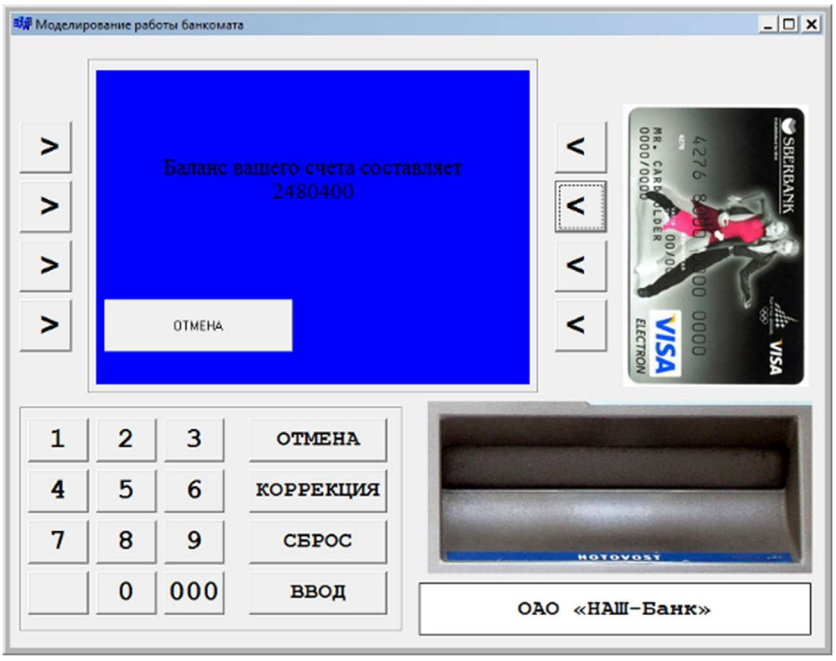

Рисунок 2.9 – Перевірка залишку на рахунку

При виборі користувачем операції проведення платежів здійснюється запит оператора стільникового зв'язку (рис. 2.10), номера телефону (рис. 2.11) та суми платежу (рис. 2.12)

Перед проведенням операції платежу проводиться перевірка наявності зазначеної суми на балансі картки. У разі наявності здійснюється повернення в основне меню (рис. 2.5). В іншому випадку виводиться повідомлення з помилкою (рис. 2.7).

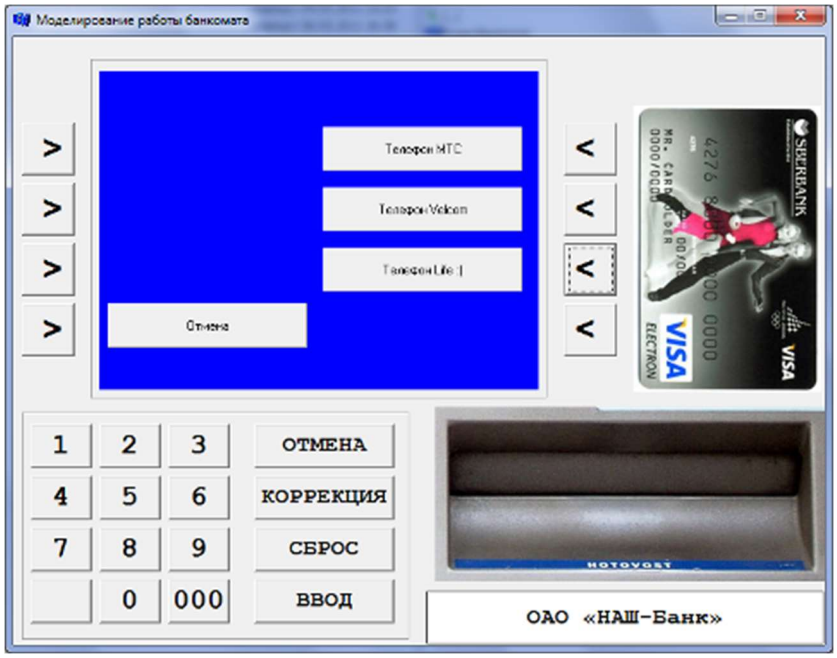

Рисунок 2.10 – Запит оператора стільникового зв'язку

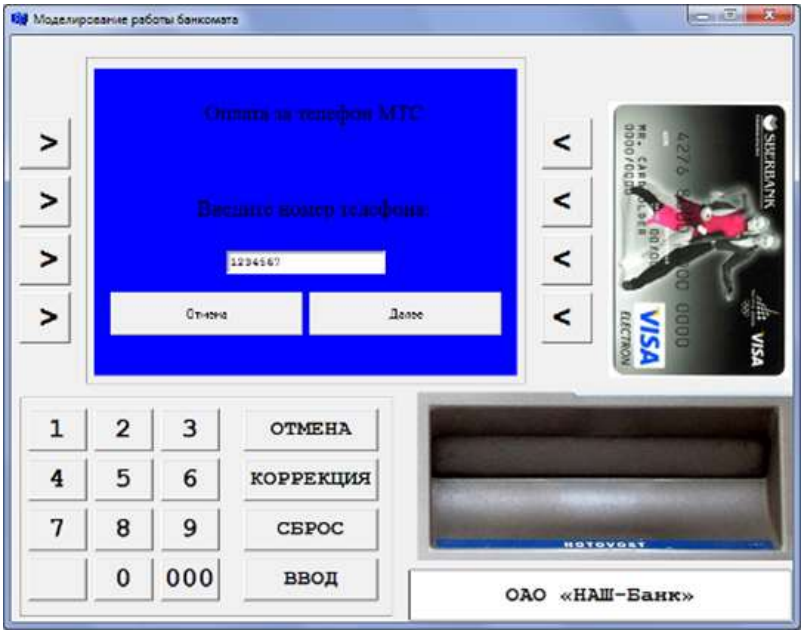

Рисунок 2.11 – Запит номеру телефона

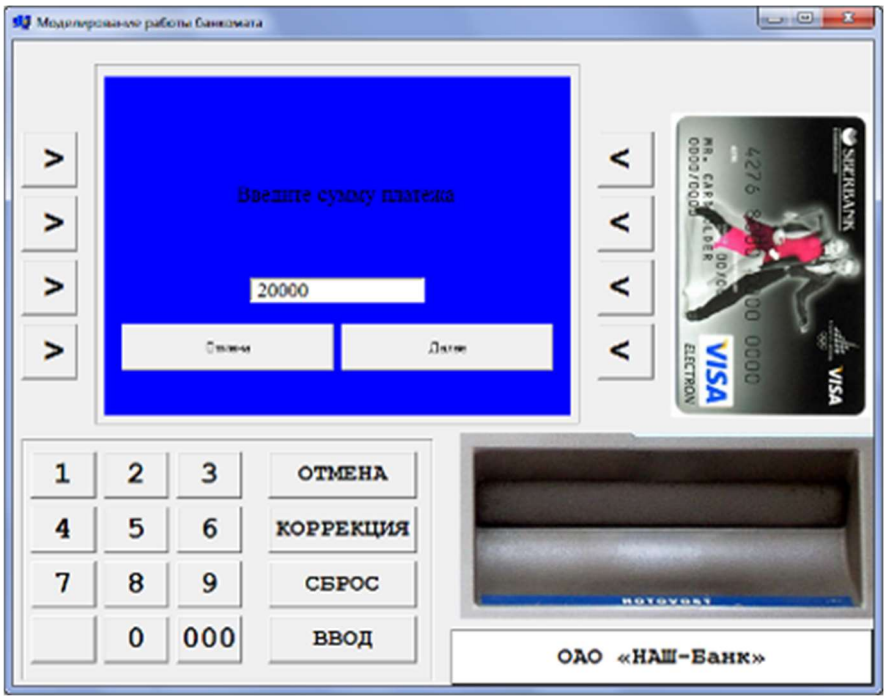

Рисунок 2.12 – Запит суми платежу

При виборі користувачем «Вихід» проводиться «витяг» картки та виведення основного вікна (рис. 2.13).

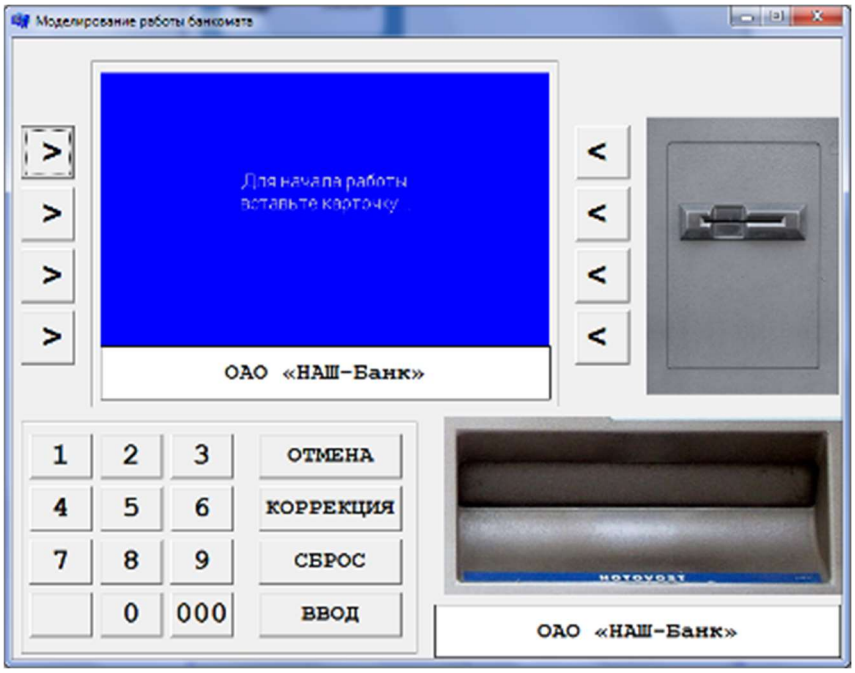

Рисунок 2.13 – Основне вікно

3 ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

1. Реалізувати програму, опис якої наведено в Розділі 2.

2. В якості мови програмування використати будь-яку мову на ваш вибір.

3. Дані в програмі повинні бути зашифровані.

4. Навести вихідний код програми.

5. Разом зі звітом надати скомпільовану програму у архіві.

Запитання для самоперевірки

1. Який мінімальний набір компонентів банкомату?

2. Що являє собою стандарт CEN/XFS?

3. Що таке стейти?

4. Як влаштований екран банкомату?