Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інтелектуальних інформаційних технологій та автоматизації
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах»

Виконав: студент 2-го курсу, групи 1КН-23м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
Загнітко В. М.
(прізвище та ініціали)

Керівник: к.т.н., проф. каф. КН
Колесницький О.К.
(прізвище та ініціали)
«05» 12 2024 р.

Опонент: к.т.н., ~~професор~~ каф. КСУ
Биков М. М.
(прізвище та ініціали)
«05» 12 2024 р.

Допущено до захисту
Завідувач кафедри КН
д.т.н., проф. Яровий А.А.
(прізвище та ініціали)
«06» 12 2024 р.

Вінниця ВНТУ - 2024 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 122 «Комп'ютерні науки»
Освітньо-професійна програма – «Системи штучного інтелекту»

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
Д.т.н., проф. Яровий А.А.
11.10 2024 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Загнітку Віктору Миколайовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах
керівник роботи к.т.н., проф. кафедри КН Колесницький О. К.
затверджені наказом вищого навчального закладу від "17" 09 2024 року № 310
2. Строк подання студентом роботи 11.11.24 2024 року
3. Вихідні дані до роботи:
Вхідні дані – кількість вихідних форм – не менше 3; програмні інструменти (мови та фреймворки); використання об'єктно-орієнтованої мови програмування; використання СУБД Microsoft SQL Server.

4. Зміст текстової частини:
Вступ, аналіз предметної області моделювання ігрових ситуацій в розподілених масштабованих системах, розробка інформаційної технології моделювання ігрової ситуації в розподілених масштабованих системах, програмна реалізація інформаційної технології моделювання ігрової ситуації в розподілених масштабованих системах, тестування та аналіз результатів роботи програми моделювання ігрової ситуації «Техаський холдем», економічна частина, висновки, перелік використаних джерел, додатки

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень)
Структура інформаційної технології моделювання ігрової ситуації «Техаський холдем», граф-схема алгоритму прийняття рішень в «Техаському холдемі», головні параметри гри в «Техаському холдемі», робочі вікна програми моделювання ігрової ситуації в розподілених масштабованих системах.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	виконання прийняв
1-4	Колесницький О.К., к.т.н., проф. каф. КН		
5	Лесько О. Й., к. е. н., проф. зав. каф. ЕПВМ		

7. Дата видачі завдання 02.09 2024 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз сучасного рівня інформаційних технологій моделювання ігрової ситуації в розподілених масштабованих системах. Постановка задач дослідження	02.09.24 - 16.09.24	
2	Побудова моделей ігрової ситуації в розподілених масштабованих системах	17.09.24 - 29.09.24	
3	Практичне застосування та оцінка ефективності розроблених моделей	30.09.24 - 28.10.24	
4	Підготовка економічної частини	29.10.24 - 01.11.24	
5	Апробація та/або впровадження результатів дослідження	02.11.24 - 04.11.24	
6	Оформлення пояснювальної записки, графічного матеріалу та презентації	05.11.24 - 11.11.24	

Студент _____ Загнітко В. М.
(підпис)

Керівник роботи _____ Колесницький О.К.
(підпис)

АНОТАЦІЯ

УДК 004.8

Загнітко В. М. Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах. Магістерська кваліфікаційна робота зі спеціальності 122 – «Комп'ютерні науки», освітня програма – «Системи штучного інтелекту». Вінниця: ВНТУ, 2024. 115 с.

На укр. мові. Бібліогр.: 21 назв; рис.: 34; табл. 10.

Дана магістерська кваліфікаційна робота присвячена розробці програмного забезпечення для інформаційної технології моделювання ігрової ситуації в розподілених масштабованих системах. Виконано аналіз сучасних програм-аналогів, які використовуються для інформаційної технології моделювання ігрової ситуації в розподілених масштабованих системах досліджено методи, які можуть бути використані для реалізації поставленої задачі. Запропоновано оціночну функцію для аналізу ситуацій, розроблено математичну модель, на основі якої побудована система прийняття рішень Розроблено алгоритм роботи програми та відповідне програмне забезпечення на мові програмування C# з програмною платформою на .NET 4.0, запити до бази даних виконано за допомогою мови LINQ. Аналіз роботи програмного забезпечення показав розширення функціональних можливостей інформаційної технології моделювання ігрової ситуації в розподілених масштабованих системах.

Графічна частина складається із 6 плакатів.

У економічному розділі доведено доцільність проведення науково-дослідної роботи за даною темою. Термін окупності становить 0,4 р., що свідчить про комерційну привабливість науково-технічної розробки.

Ключові слова: ігрова ситуація; прийняття рішень; масштабована система; розподілена система.

ABSTRACT

Zahnitko V. M. Information technology for modeling game situations in distributed scalable systems. Master's thesis in the specialty 122 - «Computer sciences», educational program – «Artificial intelligence systems». Vinnytsia: VNTU, 2024. 115 p.

In Ukrainian language. Bibliogr .: 21 titles; fig . 34; table 10.

This master's qualification work is devoted to the development of software for information technology modeling of game situations in distributed scalable systems. An analysis of modern analog programs used for information technology modeling of game situations in distributed scalable systems was performed, methods that can be used to implement the task were investigated. An evaluation function for analyzing situations was proposed, a mathematical model was developed, on the basis of which a decision-making system was built. The algorithm of the program and the corresponding software were developed in the C# programming language with a software platform on .NET 4.0, database queries were performed using the LINQ language. Analysis of the software operation showed an expansion of the functional capabilities of information technology modeling of game situations in distributed scalable systems.

The graphic part consists of 6 posters.

The economic section proves the feasibility of conducting research work on this topic. The payback period is 0.4 years, which indicates the commercial attractiveness of scientific and technical development.

Key words: game situation; decision-making; scalable system; distributed system.

ЗМІСТ

ВСТУП

Актуальність. Штучний інтелект – область досліджень і прикладних розробок, спрямованих на створення програмно-апаратних засобів, здатних до вирішення таких завдань, які передбачають застосування людиною своїх інтелектуальних здібностей, іншими словами, СШІ (система штучного інтелекту) – це інформаційна система, яка має інтелектуальні властивості. Список інтелектуальних властивостей не є одностайно визначеним, але все ж таки має більш-менш сталий перелік, наприклад: мислення, здатність до навчання, уява, сприйняття, увага, пам’ять, ерудиція. Тобто інформаційні системи, які мають перелічені вище властивості, можуть називатися інтелектуальними (або системами з елементами штучного інтелекту).

Такі системи широко застосовуються в різних галузях науки і техніки. Одним з напрямків застосування є моделювання поведінки людини, в тому числі в моделюванні систем аналізу ігрових ситуацій. На сучасному етапі розвитку технологій імітація поведінки людини в різних ситуаціях виникає при розв’язанні широкого спектру задач. В зв’язку з вищевказаним тема даної роботи є дуже актуальною і має важливе практичне значення.

Інтелектуальні якості реалізуються в інформаційній системі за допомогою певних механізмів – інтелектуальних технологій; якщо в інформаційній системі не використовуються інтелектуальні технології, вона не є інтелектуальною. Таким чином, першою найважливішою основою для класифікації СШІ є класифікація за видом інформаційної технології, яка використовується в інформаційній системі. Звісно, що в інформаційній системі можуть бути одночасно застосовані декілька інтелектуальних технологій.

В даній магістерській роботі використовуються такі інтелектуальні технології: сортування за методом Хоара, бази знань, «жадібний алгоритм», стратегії гравця в теорії ігор, теорія прийняття рішень.

Зв'язок роботи з науковими програмами, планами, темами. Магістерська робота виконана відповідно до напрямку наукових досліджень

кафедри комп'ютерних наук Вінницького національного технічного університету 22 К1 «Моделі, методи, технології та пристрої інтелектуальних інформаційних систем управління, економіки, навчання та комунікацій» та плану наукової та навчально-методичної роботи кафедри.

Мета і завдання досліджень. Метою магістерської кваліфікаційної роботи є розширення функціональних можливостей нформаційної технології моделювання ігрової ситуації «Техаський холдем».

Для досягнення мети розробки необхідно виконати такі задачі:

- провести аналіз предметної області моделювання ігрової ситуації в розподілених масштабованих системах;
- розробити форму подання ігрової ситуації в «Техаському холдемі»;
- розробити математичну модель гри «Техаський холдем»;
- розробити процес прийняття рішень в «Техаському холдемі»;
- розробити стратегію гравця в «Техаському холдемі»;
- розробити структуру інформаційної технології моделювання ігрової ситуації «Техаський холдем»;
- розробити оціночну функцію гри в «Техаському холдемі»;
- розробити алгоритм роботи програмного засобу;
- виконати програмну реалізацію запропонованої інформаційної технології;
- провести тестування програмного продукту та виконати аналіз отриманих результатів.

Об'єкт дослідження – процес комп'ютеризованого розв'язання інтелектуальних ігрових задач, які не мають чітко визначених виграшних стратегій.

Предмет дослідження – інформаційна технологія та програмні засоби для моделювання ігрової ситуації «Техаський холдем» та їх функціональні можливості.

Методи дослідження. У роботі використані наступні методи наукових досліджень: системного аналізу, теорії ігор для реалізації інформаційної

технології, методи математичного моделювання, методи математичної статистики для розробки процесу розв'язання задачі та обрахунків результатів експериментів із програмним засобом, об’єктно-орієнтованого програмування.

Наукова новизна одержаних результатів.

1. Набула подальшого розвитку інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах, яка відрізняється використанням бази знань готових рішень та «жадібного алгоритму», що дозволило розширити функціональні можливості інформаційної технології моделювання ігрової ситуації «Техаський холдем».

Практичне значення одержаних результатів полягає в тому, що на основі проведених досліджень розроблено програмне забезпечення ігрової ситуації «Техаський холдем».

Запропонована інформаційна технологія сприяє підвищенню реалістичності програмних засобів гри в «Техаський холдем», зокрема:

- розроблено алгоритм роботи програмного забезпечення гри в «Техаський холдем»;
- розроблено програмні засоби для гри в «Техаський холдем».

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується коректністю постановки завдання, коректністю використання математичного апарату методів дослідження, експериментальними дослідженнями тестування програмної реалізації інформаційної технології гри в «Техаський холдем». Адекватність розроблених математичних моделей підтверджується результатами експериментальних досліджень.

Особистий внесок здобувача. Усі результати, наведені у магістерській кваліфікаційній роботі, отримані самостійно. У працях, написаних у співавторстві, здобувачу належать: аналіз ігрової ситуації «Техаський холдем» та методів розширення функціональних можливостей програмних засобів гри в «Техаський холдем» [1,2,3].

Апробація результатів роботи. Результати роботи були апробовані на конференції «LIII науково-технічна конференція підрозділів ВНТУ», Вінниця, 20-22 березня 2024 р.».

Публікації. За результатами досліджень опубліковано троє тез доповідей на науково-технічній конференції [1,2,3]. Також було подано заявку на видачу свідоцтва про авторське право на твір-програму . Дата подачі заявки 02.12.2024.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ МОДЕЛЮВАННЯ ІГРОВОЇ СИТУАЦІЇ В РОЗПОДІЛЕНИХ МАСШТАБОВАНИХ СИСТЕМАХ

В даному розділі проведено аналіз сучасних методів моделювання систем, наведено основні правила гри «Техаський холдем», проаналізовано існуючі покер-системи та наведено постановку задачі.

1.1 Основні визначення предметної області

Масштабованість – це здатність системи нарощувати свою продуктивність зі збільшенням апаратних ресурсів системи.

Масштабованість можна оцінити через відношення приросту продуктивності системи до приросту використовуваних ресурсів. Чим ближче це відношення до одиниці, тим краще.

Розподілена система – це система, яка має кілька центрів оброблення даних, що дає змогу виконувати паралельні обчислення. Прикладом розподіленої системи може слугувати мультипроцесорний комп’ютер. Мультипроцесорний комп’ютер має декілька процесорів, кожен з яких незалежно від інших виконує свою програму, а взаємодія між ними відбувається через спільну оперативну пам’ять.

Масштабованість та розподіленість системи будемо розглядати на прикладі гри в покер «Техаський холдем».

Покер – родина азартних та спортивних карткових ігор, в яких перемагають певні комбінації гральних карт [4].

Омаха та Техаський холдем одні із найпопулярніших видів покеру, які грають на офіційному рівні.

Омаха – гра, що походить від «Техаського холдема». Кожному гравцеві роздаються чотири карти (кишенькові карти), що належать тільки йому. Також здаються п’ять відкритих загальних карт.

Техаський холдем – варіант покеру, де гравець вибирає п’ять кращих карт з двох власних і п’яти загальних для всіх гравців.

Головні визначення в «Техаському холдемі» [4]:

Бет – гравець оголошує ставку і додає фішки в банк. Такою дією гравець змушує того, хто знаходиться після нього зрівняти ставку, або спасувати.

Колл – гравець урівнює ставку, оголошену іншим гравцем перед ним і таким чином продовжує боротьбу за банк.

Чек – гравець передає слово наступному, залишаючи ставку незмінною. Така дія можлива тільки якщо до нього ставка не була зроблена.

Рейз – гравець підвищує ставку, що була підвищена до нього.

Фолд – гравець пасує (скидає свої карти), відмовляючись від подальшої боротьби за банк.

Математичне сподівання – це середня кількість фішок, які можна програти або виграти на кожній конкретній ставці.

Одси (шанси банку) – визначає математичне сподівання в ході гри.

Пот-одси (потенційні шанси банку) – шанси банку з врахуванням майбутніх ставок.

Рука – в покері комбінація, яку має на даний момент гравець.

Аут – карта, яка може покращити руку.

Комбінація – це, те що є на даний момент в гравця.

Великий блайнд – це більша з двох так званих «сліпих» ставок, які гравці зобов’язані зробити в тих видах покеру,де використовується система «сліпих» ставок, наприклад «Техаський холдем».

Малий блайнд – це менша з двох так званих «сліпих» ставок, які гравці зобов’язані зробити в тих видах покеру,де використовується система «сліпих» ставок, наприклад «Техаський холдем».

Стек – це загальна кількість фішок, яку має гравець та може ними скористатися для встановлення ставки.

Шанси на покращення – це ймовірність покращення комбінації на послідуючих стадіях гри.

1.2 Опис правил гри «Техаський холдем»

На сьогоднішній день існує безліч різновидів покеру, найпопулярніші з них «Техаський холдем» та «Омаха». Деякі граються на офіційному рівні, деякі популярні тільки в домашньому колі. Нижче наводяться коротка характеристика найпопулярніших різновидів покеру.

В Омаха кожному гравцеві роздаються чотири карти (кишенькові карти), що належать тільки йому. Також здаються п’ять відкритих загальних карт.

Кожен з гравців може використовувати рівно три з п’яти загальних карт з двома своїми кишеньковими картами, щоб зібрати найкращу комбінацію покеру з п’яти карт.

В Техаському холдемі гравець вибирає п’ять кращих карт з двох власних і п’яти загальних для всіх гравців. У «Техаському холдемі» можуть грати від 2 до 10 осіб.

У покері найчастіше використовується стандартна колода з 52 листів з рівнозначними мастями [4].

В додатку Д наведені головні комбінації для гри в покер «Техаський холдем» [5].

1.3 Аналіз сучасних покер-систем в «Техаському холдемі»

Проведемо аналіз найпоширеніших сучасних покер систем.

Покер-система – це система, яка надає можливість грати гравцеві в покер. Покер-система немає чітко визначеного алгоритму наперед, саме тому в покер-системах використовується штучний інтелект.

Найбільш популярними покер-системами на момент написання роботи є: pokerstars, cristalpalaceonline, erapokera, lasvilis, але найбільшу популярність має pokerstars, через те, що там простий візуальний інтерфейс та система навчається в процесі її використання.

На сьогоднішній день існує тисячі різних покер-систем, але покер-система це теж програма, яка моделює ігрову ситуацію. Як будь-яка програма, так і кожна покер-система має свої недоліки та переваги, які грають дуже важливу роль, тому що від цього залежить кількість людей, які надають перевагу саме даній покер-системі, від чого зростає рейтинг сайту і самої покер-системи. Звичайно, рейтинг покер-системи обернено пропорційний алгебраїчній сумі недоліків даної системи, іншими словами чим менше недоліків має покер-система тим більше їй будуть надавати переваги гравці.

Недоліки та переваги кожної з покер-систем, наведені в таблиці 1.1.

Таблиця 1.1 – Переваги та недоліки покер-систем

Покер-система	Переваги покер-системи	Недоліки покер-системи
pokerstars	Простий візуальний інтерфейс. Система навчається в процесі її використання.	Дуже складний алгоритм, через що система працює досить довго.
erapokera	Система навчається в процесі її використання. Простий візуальний інтерфейс.	Дуже складний алгоритм, через що система працює досить довго.
lasvilis	Проста в використанні та досить швидка порівняно з іншими покер система.	Система не завжди знаходить оптимальне рішення на кожному етапі гри.
cristalpalace	Швидка в використанні. Досить велика кількість людей надає перевагу даній покер системі.	Немає нейронної мережі, через що система не навчається, складний візуальний інтерфейс.

Запам’ятовування гри – це збереження різних ситуацій, в якій побував гравець з опонентом, іншими словами система, яка не володіє даною можливістю, не враховує психологію опонента.

Багато ігор, які моделюють гру справжнього опонента перебирають різні можливі комбінації, щодо подальшого ходу. Звісно перебір різних варіантів знайде найкращий подальший хід, але перебір цих варіантів є дуже трудомісткою операцією, яка займає достатньо великий час.

Тому рекомендується створити систему, яка не буде перебирати усі можливі варіанти подальшого ходу, а буде обирати хід на основі вже готових рішень, які в свою чергу вже містять оптимальні варіанти. Розпаралелення обчислень підвищить швидкодію за рахунок виконання паралельних обчислень, які буде виконувати масштабована система.

1.4 Постановка задачі

1. Вхідними даними є малий (великий) блайнд, режим гри звичайний (професійний), стек гравця (опонента). В процесі гри на гральний стіл випадають карти з даних карт формується комбінація, яка в подальшому буде використовуватися для визначення шансів гравця (опонента).

2. Системні вимоги до розроблюваної гри:

2.1 В комп’ютері повинен бути встановлений процесор по характеристикам не нижче AMD Ryzen 3600 2.9 MHZ або Intel Core i5-10400F;

2.2 Об’єм оперативної пам’яті (RAM) не менше 1024 мб;

2.3 Об’єм відеопам’яті (VRAM) не менше 256 мб.

3. В якості вихідних даних виступає готова комбінація з карт, шанси гравця (опонента), по складеній комбінації визначається переможець партії.

1.5 Висновок до розділу 1

В даному розділі було проведено аналіз предметної області інформаційних технологій моделювання ігрової ситуації в розподілених масштабованих системах, наведено основні визначення предметної області, проаналізовано і подано головні правила гри в «Техаському холдемі», проведено аналіз існуючих покер-систем в «Техаському холдемі» (pokerstars, cristalpalaceonline, erapokera, lasvilis), наведено їх переваги та недоліки. В постановці задачі наведено вхідні, вихідні дані, а також наведені системні вимоги до розроблюваної гри. На основі аналізу недоліків сучасних покер-систем сформульовано мету роботи – розширення функціональних можливостей.

2 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ МОДЕЛЮВАННЯ ІГРОВОЇ СИТУАЦІЇ «ТЕХАСЬКИЙ ХОЛДЕМ» В РОЗПОДІЛЕНИХ МАСШТАБОВАНИХ СИСТЕМАХ

В даному розділі наведено обґрунтування форми подання ігрової ситуації в «Техаському холдемі». Розроблена математична модель гри «Техаський холдем», проведено аналіз задачі прийняття рішень в «Техаському холдемі», проаналізовано стратегію в «Техаському холдемі», обчислено ймовірнісні характеристики в різних ігрових ситуаціях.

2.1 Розробка форми подання ігрової ситуації в «Техаському холдемі»

Від вибору представлення задачі напряму залежить швидкодія програми та об’єм коду. Швидкодія в програмах має велике значення, так як кінцеві користувачі програми віддають перевагу програмам, які швидше працюють. Тому метод представлення задачі грає надзвичайно велике значення в проектуванні програми [6].

Одним із ключових атрибутів гри в покер є карти. Для їх представлення використовується структура, яка буде включати в себе два відкритих поля:

1. Буквене поле;
2. Ранг карти.

Буквене поле в структурі позначає масть карти, а ранг карти в структурі складається з однієї букви чи цифри більше одиниці, яка буде вказувати на ранг цієї карти по відношенню до інших. Приклад представлення карти 3♣ (трійка трефи) в структурі, рисунок 2.1.

Коли карта 3♣ (трійка трефи) прийде на вхід структури вона запишеться як «С3», буква «С» буде записана в «буквене поле» структури, а «3» запишеться в поле «ранг карти».

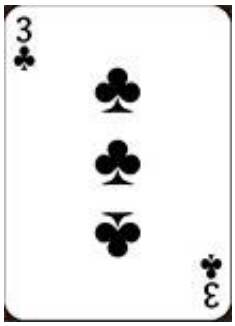


Рисунок 2.1 – Трійка трефи

Кожна карта в колоді може бути однією з 4 мастів, назви мастів карт, які будуть використані при розробці програми такі:

1. Spades (S) – піки;
2. Hearts (H) – чирви;
3. Diammonds (D) – бубни;
4. Clubs (C) – трефи.

Множина рангів карт представлена в програмі множиною 2.1:

$$D = \{ \text{'2', '3', '4', '5', '6', '7', '8', '9', 'T', 'J', 'Q', 'K', 'A'} \}, \quad (2.1)$$

де '2', '3'…'K', 'A' – ранги карт.

Карта з рангом «10» в програмі буде представлятися як латинська буква «T», тому що в ASCII таблиці немає такої цифри як 10, ця цифра кодується двома байтами, тому в програмі вона буде представлена як латинська буква «T».

В об’єктно-орієнтованому програмуванні структура представляється як об’єкт, це означає, що можна створити колекцію однотипних елементів (масив) [7]. Для представлення карт будемо використовувати масив об’єктів структури, яка буде тасуватися після кожної зіграної партії. Масив буде містити 52 елементи, так як для гри в покер «Техаський холдем» використовується саме 52 карти. Приклад розміщення декількох карт в масиві об’єктів, представляється множиною 2.2:

$$Q = \{"S2","S3","S4"\ldots"SA"\ldots"CA"\}, \quad (2.2)$$

де "S2","S3"…"CA" – розміщення карт в масиві об’єктів.

Рука опонента та рука гравця будуть представлятися як стрічки, наприклад: «S2 C4», рисунок 2.2.

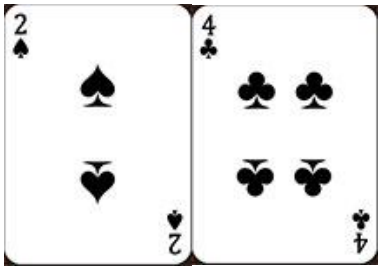


Рисунок 2.2 – Представлення руки «S2 C4» в програмі

Стрічка «S2 C4» інформує гравця, що його кишенькові карти є пікова двійка і трефова четвірка. Аналогічно буде представлятися рука опонента.

Флоп, терн і рівер також будуть представлятися як стрічки, кожна з яких буде містити по 8, 11, та 14 елементів відповідно. Для флопа стрічка буде містити 8, тому що 2 елемента йде на представлення самої карти та один елемент на пропуск. Цю залежність можна описати наступною формулою 2.3:

$$3 * n - 1, \quad (2.3)$$

де n – це кількість карт, що потрібно представити в стрічці.

Важливою стадією проектування математичної моделі покеру є складання комбінацій. Як було описано вище комбінацій в покері існує тільки 10, а саме: Флеш-рояль, Стріт-флеш, Каре, Фул-хаус, Флеш, Стріт, Сет, Дві пари, Пара і Кікер. Для складання комбінацій буде створено окремий клас,

який буде інкапсулювати код, який необхідний для складання комбінацій. Даний клас буде містити тільки один відкритий метод, всі інші будуть як private (закриті), тобто за межами класу дані члени класу будуть недоступними. Для сортування карт будемо використовувати сортування методом Хоара, дане сортування має складність O($\log_2 n^n$), що в свою чергу забезпечить швидке сортування карт [8]. Сортування необхідне для спадаючої послідовності рангів карт. Так як сортування методом Хоара застосовується лише до чисел, то вводиться масив індексів. Масив індексів формується таким чином:

1. Застосовується операція конкатенації до руки та флопу
2. Створюється масив, де будуть міститися ранги карт в зростаючому порядку, див. множину 2.1.
3. Запустити цикл по новоутвореній стрічці, на кожній ітерації якого порівнюється ранг карти з рангами карт, що містяться у масиві;
4. Зберегти індекси, що отримано на кроці 3 в новий масив;
5. Сортується масив індексів методом Хоара;
6. На останньому кроці запускається два цикли, де порівнюються, ранги відсортованих індексів карт з стрічкою, яка була утворена на кроці 1, та зберігається нова послідовність карт (відсортованих) в новоутвореній стрічці.

Це забезпечить легше складання комбінацій карт. Наприклад, якщо в метод передано дві стрічки (рука та флоп), де рука: «S4 H6», рисунок 2.3, а на флоп прийшли такі карти :«C4 SA D6», рисунок 2.4.

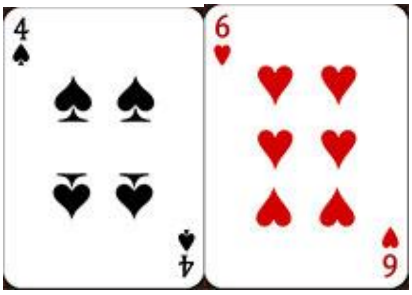


Рисунок 2.3 – Рука «S4 H6»

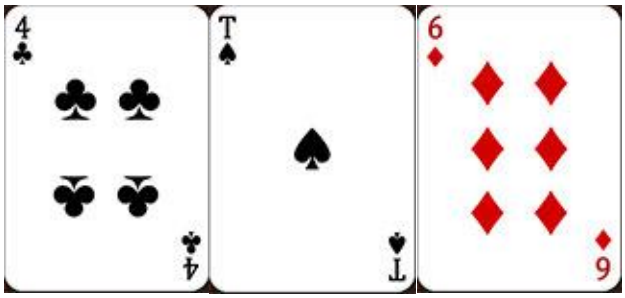


Рисунок 2.4 – Флоп «C4 SA D6»

В результаті застосування всіх вище наведених кроків виходить відсортована стрічка: «SA D6 H6 C4 S4», рисунок 2.5.

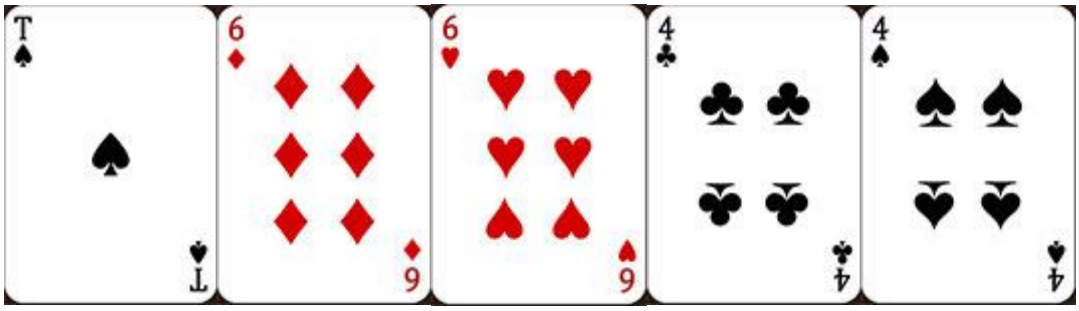


Рисунок 2.5 – Відсортована стрічка «SA D6 H6 C4 S4»

Застосувавши операцію сортування необхідно перевірити чи немає готової комбінації [8]. Для цього необхідно враховувати кількість входжень карти даного рангу (якщо не шукаємо, Флеш, Стріт-флеш чи Флеш-рояль) в стрічку, щоб це реалізувати потрібно використовувати метод, який буде шукати кількість входжень карти даного рангу в стрічку. Наприклад для стрічки: «SA D6 H6 C4 S4», див. рисунок 2.5, програма згенерує такі результати: «1, 2, 2, 2, 2». «1», тому що піковий туз «SA» входить в задану стрічку лише один раз, всі інші елементи входять двічі, рисунок 2.6.

Для інших комбінацій, таких як: «Флеш», «Стріт-флеш» чи «Флеш-рояль» потрібно знаходити кількість входжень не рангів карт, а кількість входжень мастів карт в стрічку. «Стріт-флеш» та «Флеш-рояль» окрім знаходження флешу потребує також впорядкування карт (тобто стріта), для

«Флеш-роялю» оверкарта стріта буде «Туз», для «Стріт-флеша» будь-яка, тільки не «Туз».

Рисунок 2.6 – Застосування операції «Кількість входжень» до стрічки «SA D6 H6 C4 S4»

Щоб представити стек (кількість фішок, які має на даний момент гравець) використовується структура, яка буде містити два поля [6]:

1. Стек опонента (комп’ютера);
2. Стек гравця;

Ставки обох гравців будуть міститися в змінній, яка буде занулюватися при кожному завантаженню гри.

Кожна ставка, яку буде робити гравець чи комп’ютер буде представлятися якоюсь певною кількістю фішок. Щоб визначити, які фішки необхідні для представлення даної ставки буде використовуватися «Жадібний алгоритм». Визначивши, які саме фішки необхідні для представлення ставки від стеку гравця чи опонента буде відніматися ставка, тобто, якщо стек складає

1000, а ставка 200, то після того, як будуть визначені фішки необхідні для представлення даної ставки стек стане 800.

Математичне сподівання приходу необхідної карти – це ймовірність приходу карти без врахування руки опонента. Рука опонента не враховується, тому що підрахування математичного очікування проводиться тільки на базі тієї інформації, яка є в гравця, а це тільки інформація про його руку та про карти на столі. Саме через це виконуються обчислення, якби гравець був єдиним гравцем за гральним столом.

2.2 Розробка математичної моделі гри в «Техаському холдемі»

Для розробки математичної моделі гри визначимо основні математичні залежності, які дозволять обґрунтувати доцільність певного ходу в кожній ситуації.

Математична модель представлення задачі відіграє одну із найважливіших стадій проектування розроблюваного програмного забезпечення. Оптимальне представлення математичної моделі не тільки забезпечить більш оптимальну стратегію опонента (комп’ютера), а й може покращити швидкодію програми, що є одним із найважливіших критеріїв вибору програми кінцевому користувачу.

Найефективніший метод розв’язку поставленої задачі є представлення кожної ігрової ситуації у вигляді математичних формул, де на кожній стадії гри буде оцінюватися кожна ігрова ситуація за допомогою оціночної функції.

Оціночна функція в «Техаському холдемі» – допомагає визначити доцільність ходу, на кожній стадії гри. На кожному етапі гри оціночна функція буде розраховуватися згідно нижче наведених формул:

1. Шанси на покращення розраховується за формулою 2.4 [5]:

$$S = \frac{O}{K-O}, \tag{2.4}$$

де O – карти, які можуть покращити комбінацію,

K – кількість карт в колоді, не рахуючи карти опонента.

2. Щоб визначити доцільність колу, рахується можливий програш та можливий виграш за даної ставки, розрахунки проводяться за формулами 2.5 – 2.9 [5]:

$$W = O * B + H, \tag{2.5}$$

де O – карти, які можуть покращити комбінацію,

B – кількість фішок, що містяться в банку,

H – сила комбінації, що має опонент (комп’ютер)

$$L = (K - O) * R, \tag{2.6}$$

де O – карти, які можуть покращити комбінацію,

K – кількість карт, що залишилися в колоді,

R – ставка, яку поставив опонент.

Щоб визначити чи доцільно зрівнювати ставку проводяться розрахунки за формулою 2.7 [5]:

$$\text{if } (W - L) > 0, \tag{2.7}$$

де W – можливий виграш;

L – можливий програш.

В такому випадку краще робити порівнювати ставку (колл), тому що можливий виграш при даній ставці більший ніж програш.

$$\text{if } (W - L) \gg 0, \tag{2.8}$$

де W – можливий виграш;

L – можливий програш.

В такому випадку краще підвищувати ставку (рейз), тому що можливий виграш при даній ставці набагато більший ніж програш. Набагато більший – це можливий виграш більший приблизно в 1.5 за можливий програш [9]:

$$\text{if } (W - L) < 0, \tag{2.9}$$

де W – можливий виграш;

L – можливий програш.

В такому випадку краще скидати карти (фолд), тому що можливий програш більший ніж виграш.

3. В покері математичне сподівання відіграє важливу роль, математичне сподівання розраховується за формулою 2.10 [10]:

$$E = \frac{O}{K} \tag{2.10}$$

де O – карти, які можуть покращити комбінацію,

K – кількість карт, що залишилися в колоді.

4. Актуальність ставки визначає приблизно розмір ставки, яка відповідає ігровій ситуації гравця, актуальність ставки розраховується за формулою 2.11 [10]:

$$A = S * H * E, \tag{2.11}$$

де E – математичне сподівання,

H – сила комбінації, що має опонент (комп’ютер),

S – ймовірність покращення комбінації.

5. Потенційні шанси банку, розраховуються за формулою 2.12 [10]:

$$PO = \frac{R}{B}, \tag{2.12}$$

де R – ставку, яку поставив опонент,

B – кількість фішок, що містяться в банку.

Кожна із наведених формул надасть можливість опоненту (комп’ютеру) приймати найбільш оптимальне значення на кожній стадії гри.

2.3 Розробка процесу прийняття рішень в «Техаському холдемі»

Визначивши комбінацію, яку має гравець чи опонент, обираються найкращі 5 карт [5]. На флопі немає необхідності визначати найкращі 5 карт, так як кожен із гравців має по 5 карт, тобто три карти флопа плюс дві карти руки. На послідуючих стадіях гри (терн, рівер) відкидається по одній (для терна) чи по дві (для рівера) карти, для складання комбінації, або аналізу 5 найкращих карт, щоб визначити, яку комбінацію можна зібрати на послідуючих стадіях гри.

Якщо комбінацію не можливо скласти з 5 найкращих карт, тоді використовується таблиця 2.1 в якій подано ймовірності побудови покерних комбінацій [9].

Таблиця 2.1 – Ймовірності побудови покерних комбінацій

Комбінація	Комбінація, яку можна зібрати	Аути	Флоп → Терн	Терн → Рівер	Флоп → Рівер
Стріт дро плюс флеш дро	Стріт або флеш	15	2,13%	2,07%	0,85%
Флеш дро	Флеш	9	4,22%	4,11%	1,86%
Двухстороннє стріт дро	Стріт	8	4,88%	4,75%	2,18%
Сет	Фул-хаус, каре	7	5,71%	3,6%	1,99%
Немає комбінації	Пара	6	6,83%	6,67%	3,14%
Одна пара	Дві пари або сет	6	8,4%	8,2%	3,91%
Дві пари	Фул-хаус	4	10,8%	10,5%	5,07%
Стріт дро	Стріт	4	10,8%	10,5%	5,07%
Пара	Дві пари	3	5,84%	8,2%	3,91%
Сет	Каре, фул-хаус	7	5,71%	3,6%	1,99%

Комбінація, яку можна зібрати передбачає, те що, якщо випаде один із аутів цієї комбінації, тоді буде отримана відповідна комбінація, що стоїть в колонці «Комбінація». Аути – кількість карт, які можуть покращити руку. «Флоп → Терн» ймовірність появи карти на терні у відсотках, «Терн → Рівер» ймовірність появи карти на рівері у відсотках, «Флоп → Рівер» дана колонка використовується, коли гравець йде в «Олл-ін» (ставка, яка включає весь стек гравця).

2.4 Розробка стратегії гравця в «Техаському холдемі»

В покері математичне сподівання має дуже важливе значення, так як воно дає зрозуміти чи вигідна ситуація чи ні. Окрім математики в покері дуже важливу роль грає психологія, так як, знаючи свого опонента можна визначити як цей опонент поведе себе в тій чи іншій ситуації [11].

Для більш оптимального рішення потрібно підрахувати математичне сподівання для кожної із можливих комбінацій з урахуванням тієї інформації, якою володіє гравець, тобто карти, які є в руці гравця, та які прийшли на стіл, математичне сподівання ймовірності появи кожної із можливих комбінацій на терні і на рівері, підраховування можливого програшу та виграшу [12]. На базі цієї інформації можна розрахувати шанси на виграш, шанси та потенційні шанси банку. Це забезпечить більш оптимальну стратегію для обох гравців.

Математичне сподівання для окремої руки в «Техаському Холдемі» – це ймовірність складання гарної комбінації. Наприклад, якщо дві кишенькові карти гравця червової масті, і на флопі прийшли дві карти тієї ж масті, шанси банку для руки гравця на складання флеша становить 2:1. Це означає, що приблизно з кожних трьох разів розіграних партій, гравець зможе скласти флеш. Щоб розрахувати математичне очікування для руки гравця, потрібно в першу чергу знати, скільки в колоді залишилося карт, які можуть її поліпшити. Якщо кишенькові карти А ♠-К ♠ («Туз піковий» та «Король піковий»), і на флопі прийшли 2 пікові карти, то в колоді залишилися ще 9 пікових карт (тому що в колоді по 13 карт кожної масті), рисунок 2.7. Це означає, що в колоді залишилися 9 карт, за допомогою яких гравець може скласти флеш.

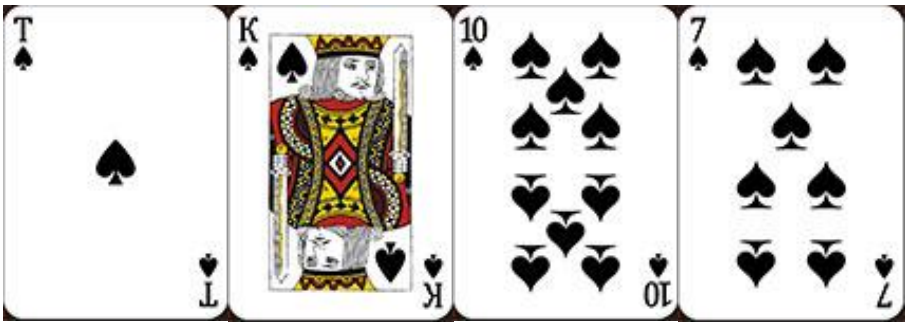


Рисунок 2.7 – Приклад ігрової ситуації в Техаському холдемі

На основі попередньо викладеної інформації можна обчислити відсоток ймовірності складання комбінації на рівері. Ймовірність легко розрахувати для одного окремого випадку наприклад, для карти, яка прийде на рівері після терну, див. формулу 2.10.

На префлопі приймається рішення згідно таблиць малих та великих бланйдів, дані таблиці наведені в підрозділі 2.5 [9].

2.5 Формування таблиці великих та малих блайндів на префлопі

Таблиці великих та малих блайндів на префлопі дають змогу визначити по кишеньковим картам ситуацію у якій знаходиться гравець на префлопі. Дані таблиці складаються з семидисяти двох кортежів, та трьох атрибутів. «$The\ fact\ that\ we\ have$» (те, що маємо) в даному домені міститься можливі комбінації карт на префлопі, «$Was\ not\ a\ raise$» (не було рейзу), не було підвищення ставки, та «$There\ are\ was\ a\ raise$» (був рейз), було підвищення ставки. На рисунку 2.8 подана таблиця великих блайндів на префлопі, на рисунку 2.9 подана таблиця малих блайндів на префлопі.

Таблиці, які подані на рисунку 2.8 та 2.9 будуть міститися в базі даних MS SQL Server’а, на префлопі буде виконуватися запит за допомогою мови LINQ до цих таблиць. До таблиць будуть передаватися кишенькові карти і наявність рейзу з таблиці буде повертатися одна буква.

ID	Was_not_a_raise	There_was_a_r...	The_fact_that_...
1	R	R	AA
2	R	R	KK
3	R	R	QQ
4	R	R	JJ
5	R	R	TT
6	R	C	99
7	C	C	88
8	C	C	77
9	C	C	66
10	C	C	55
11	C	C	44
12	C	C	33
13	C	C	22
14	R	R	AQs
15	R	R	AJs
16	R	C	ATs
17	C	C	A9s
18	C	C	A8s
19	C	C	A7s
20	C	C	A6s
21	C	C	A5s
22	C	C	A4s
23	C	C	A3s
24	C	C	A2s
25	R	R	AK
26	C	C	AQ
27	C	F	AJ
28	C	F	AT
29	R	R	KQs
30	R	C	KJs
31	C	C	KTs
32	C	F	K9s
33	C	F	K8s
34	C	F	K7s
35	C	F	K6s
36	C	F	K5s

ID	Was_not_a_raise	There_was_a_r...	The_fact_that_...
37	C	F	K4s
38	C	F	K3s
39	C	F	K2s
40	C	F	KQ
41	C	F	KJ
42	C	F	KT
43	C	F	QJs
44	C	F	QTs
45	C	F	Q9s
46	C	F	Q8s
47	C	F	QJ
48	C	F	QT
49	C	F	JTs
50	F	F	J9s
51	F	F	J8s
52	F	F	J7s
53	F	F	T9s
54	F	F	98s
55	F	F	87s
56	F	F	86s
57	F	F	85s
58	F	F	84s
59	F	F	83s
60	F	F	82s
61	F	F	77s
62	F	F	76s
63	F	F	K7s
64	F	F	Q7s
65	F	F	T7s
66	F	F	97s
67	F	F	76s
68	F	F	75s
69	F	F	74s
70	F	F	73s
71	F	F	72s
72	F	F	T6S

Рисунок 2.8 – Таблиця великих блайндів на префлопі

Якщо гравцю потрібно поставити великий блайнд і його кишенькові карти K5s (буква «s» в таблиці означає, що карти різномасті, якщо букви не вказано, то карти одномасні), згідно таблиці, якщо не було рейзу, то потрібно робити зрівняти ставку (колл), якщо був рейз (підвищення ставки), то карти краще скинути (фолд).

ID	Was_not_a_raise	There_was_a_raise	The_fact_that_we_have
1	R	R	AA
2	R	R	KK
3	R	R	QQ
4	R	R	JJ
5	R	R	TT
6	R	C	99
7	C	C	88
8	C	C	77
9	C	C	66
10	C	C	55
11	C	C	44
12	C	C	33
13	C	C	22
14	R	R	AQs
15	R	R	AJs
16	R	C	ATs
17	C	F	A9s
18	C	F	A8s
19	C	F	A7s
20	C	F	A6s
21	C	F	A5s
22	C	F	A4s
23	C	F	A3s
24	C	F	A2s
25	R	R	AK
26	R	C	AQ
27	C	F	AJ
28	C	F	AT
29	R	R	KQs
30	R	C	KJs
31	C	C	KTs
32	C	F	K9s
33	C	F	K8s
34	C	F	K7s
35	C	F	K6s
36	C	F	K5s

ID	Was_not_a_raise	There_was_a_raise	The_fact_that_we_have
37	C	F	K4s
38	C	F	K3s
39	C	F	K2s
40	C	F	KQ
41	C	F	KJ
42	C	F	KT
43	C	F	QJs
44	C	F	QTs
45	C	F	Q9s
46	C	F	Q8s
47	C	C	QJ
48	C	C	QT
49	C	C	JTs
50	F	F	J9s
51	F	F	J8s
52	F	F	J7s
53	F	F	T9s
54	F	F	98s
55	F	F	87s
56	F	F	86s
57	F	F	85s
58	F	F	84s
59	F	F	83s
60	F	F	82s
61	F	F	77s
62	F	F	76s
63	F	F	K7s
64	F	F	Q7s
65	F	F	T7s
66	F	F	97s
67	F	F	76s
68	F	F	75s
69	F	F	74s
70	F	F	73s
71	F	F	72s
72	F	F	T6S

Рисунок 2.9 – Таблиця малих блайндів на префлопі

Буква, яка повертається з таблиці, може бути представлена множиною 2.13:

$$Z = \{‘R’,’C’,’F’\}, \tag{2.13}$$

де ‘R’ – підвищувати ставку;

‘C’ – зрівнювати ставку;

‘F’ – скинути карти.

2.6 Структура інформаційної технології моделювання ігрової ситуації «Техаський холдем»

Структура інформаційної технології моделювання ігрової ситуації «Техаський холдем» представляється класами, які потрібно розробити. Кожному класу відводяться дії, які він має виконувати, ці дії відзначені в прямокутниках.

На рисунку 2.10 подана структура інформаційної технології моделювання ігрової ситуації «Техаський холдем».

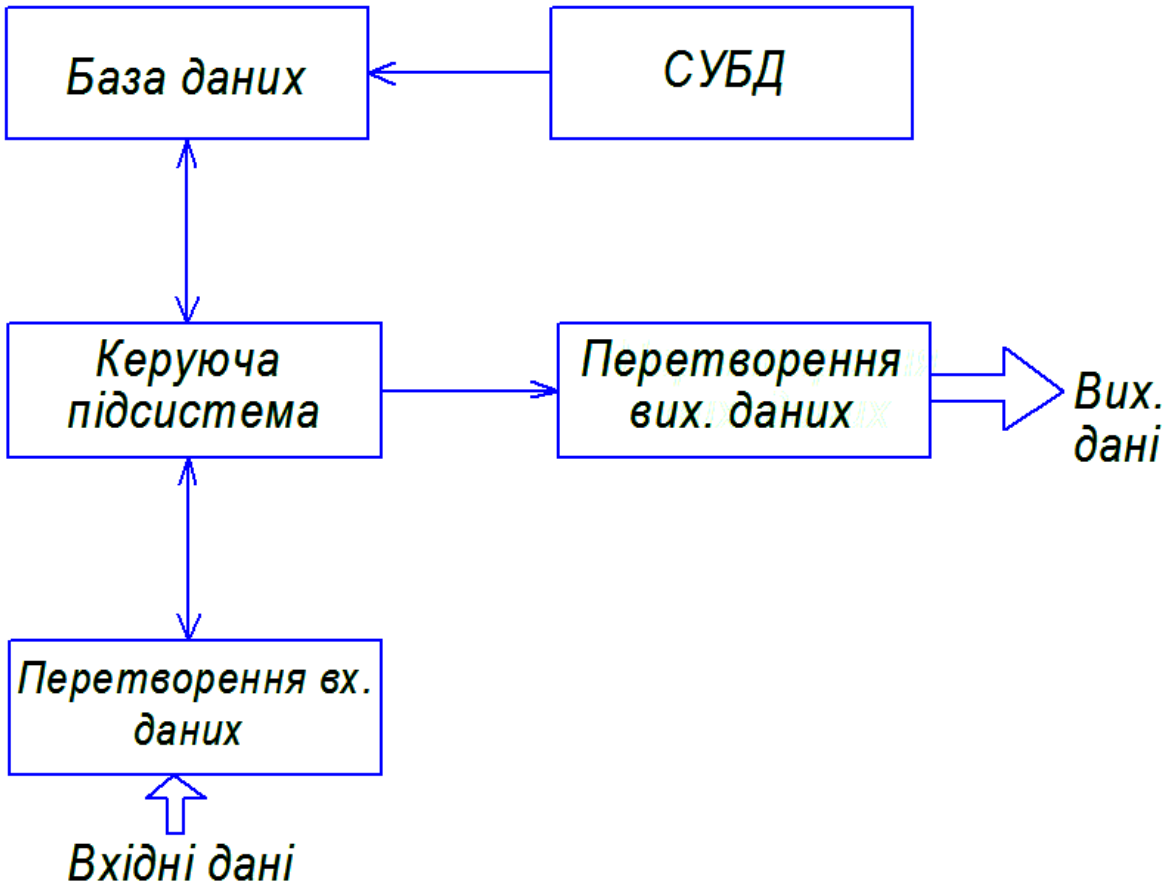


Рисунок 2.10 – Структура інформаційної технології моделювання ігрової ситуації «Техаський холдем»

Головною підсистемою в даній структурі виступає «Керуюча підсистема», вона управляє перетвореними вхідними даними, базою даних та

вихідними даними. Саме в цьому блоці відбуваються всі головні дії по роботі з програмою, а саме:

1. Сортування карт.
2. Підрахування шансів гравця та опонента (комп’ютера).
3. Створення хеш-таблиці.
4. Виконання запиту до БД.
5. Складання комбінації.
6. Визначення переможця гри.

2.7 Висновок до розділу 2

В даному розділі подано обґрунтування форми подання ігрової ситуації в «Техаському холдемі». В якості подання карт було обрано структуру, яка буде містити два поля масть та ранг карти. Така форма подання карт значно зменшить аналітичні розрахунки та спростить написання програми [7]. Для сортування колоди карт обрано метод Хоара, даний алгоритм забезпечить високу швидкість сортування карт ($\log_2 n^n$). Розроблено математичну модель гри та розроблено процес прийняття рішень в «Техаському холдемі». Запропоновано стратегію гравця в «Техаському холдемі». Стратегія гравця будується на таблицях (певний аналог бази знань), де зазначені можливі ситуації та можливі дії в них, дані таблиці забезпечують доцільність кожного наступного ходу.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ МОДЕЛЮВАННЯ ІГРОВОЇ СИТУАЦІЇ «ТЕХАСЬКИЙ ХОЛДЕМ» В РОЗПОДІЛЕНИХ МАСШТАБОВАНИХ СИСТЕМАХ

В даному розділі буде розроблено оціночну функцію [13] для гри в «Техаський холдем». Оціночна функція буде будуватися на основі математичної моделі, яка запропонована в розділі 2. Наведено приклад обчислення оціночної функції. Буде наведено приклад ігрової ситуації та повністю описано принцип роботи програми (від подання вхідних даних до отримання вихідних даних).

3.1 Розробка оціночної функції гри в «Техаському холдемі»

На префлопі приймається рішення згідно таблиць блайндів, див. рисунок 2.8 та 2.9. Дані таблиці спрощують алгоритм роботи програми, так як в них визначені наперед можливі комбінації та подальші дії. Якщо опонент (комп’ютер) чи гравець не скинули карти, роздається флоп. На флопі, терні і рівері обчислюється оціночна функція. Згідно оціночної функції приймається рішення для опонента (комп’ютера), рішення може бути «Чек», «Колл» або «Фолд». Якщо обрано «Колл», то підраховується значення, яке доцільно ставити при даній комбінації.

Оціночна функція реалізована у вигляді класу, який буде інкапсулювати методи для підрахування оціночної функції. Оціночна функція складається з математичного сподівання (середня кількість грошей, які можна програти або виграти на кожній конкретній ставці), шансів на покращення, сили руки, яку має опонент (комп’ютер), доцільність колу (підраховується можливий виграш і програш при даній ставці), потенційні шанси банку та актуальності ставки (добуток шансів на покращення на сили руки, що має опонент та на математичне сподівання). Оціночна функція представляється формулою 3.1:

$$F = S + H + C + E, \quad (3.1)$$

де S – шанси на покращення комбінації гравця;

H – сила руки гравця, по відношенню до інших комбінацій;

C – (скорочено від Call) визначає доцільність колу, при цьому визначає можливий виграш та програш при даній комбінації;

E – математичне сподівання приходу потрібної карти.

Наприклад опонент (комп'ютер) має такі кишенькові карти «CJ CK», рисунок 3.1:

Рисунок 3.1 – Кишенькові карти опонента (комп'ютера) «CJ CK»

На флоп прийшли такі карти «HQ H2 C9», рисунок 3.2:

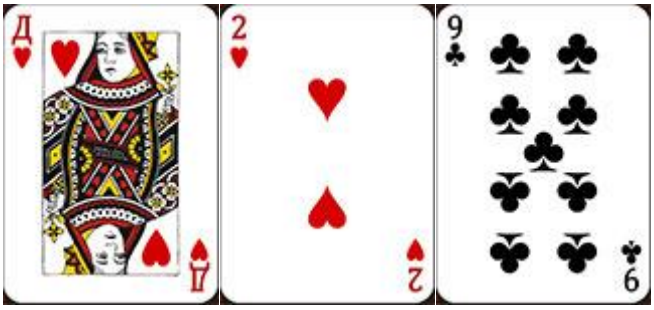


Рисунок 3.2 – Карти флопа «HQ H2 C9»

Опонент (комп'ютер) має стріт-дро («CJ CK HQ C9»), це не готова комбінація, але якщо випаде на терні чи на рівері одна із 4 десяток, опонент буде мати достатньому сильну руку для виграшу. Для даного прикладу оціночна функція буде такою:

1. Шанси на покращення: $S = 0.15$;
2. Сила руки: $H = 130$;
3. C – приймає значення з множини $\{true, false\}$, яке підраховує кількість програшу і виграшу з даною ставкою, дане значення використовується в умові;
4. Математичне сподівання: $E = 12.77$.

Оціночна функція для наведеного прикладу:

$$C = true;$$
$$F = 0.15 + 130 + 12.77 = 142.92.$$

Математичне сподівання може приймати значення як додатні так і від'ємні, в разі, якщо C дорівнює $false$, або математичне сподівання від'ємне, або, якщо умова перевірки ($S + E < PO \;\&\&\; C$), то «Чек», якщо користувач поставив ставку і доцільність колу або математичне сподівання (E) менше нуля, то «Фолд», інакше, якщо користувач поставив ставку, а ($S + E >> PO \;\&\&\; C$), то «Рейз».

Оціночна функція включає в себе всі головні парамерти, які необхідні для прийняття оптимального рішення [14] на кожній стадії гри, що в свою чергу забезпечить гравцеві сильного опонента.

3.2 Приклад ігрової ситуації в «Техаському холдемі»

1. Наприклад, рука гравця «CA DJ» (туз трефи і валет бубни), рис. 3.3.

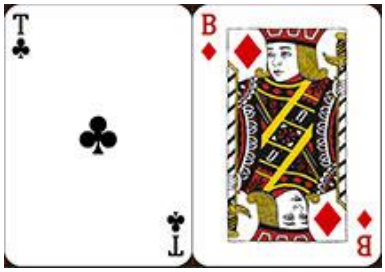


Рисунок 3.3 – Рука гравця «CA DJ»

А на флоп прийшла комбінація «DA C2 ST» (туз бубни, двійка трефи і десятка піки), рисунок 3.4.

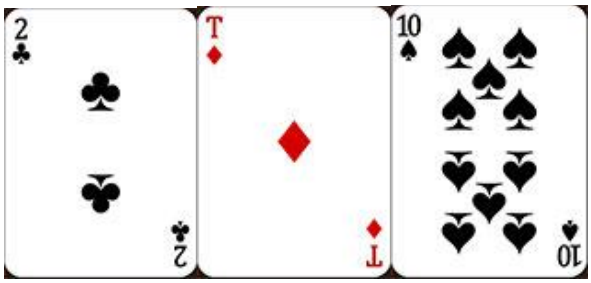


Рисунок 3.4 – Карти флопа «DA C2 ST»

2. Карти «CA DJ» передаються в метод для впорядкування їх за спаданням (від більшої до меншої). Новостворена стрічка та стадія гри (великий блайнд чи маленький блайнд) передається в LINQ запит, щоб дістати дані, щодо карт, які прийшли на префлоп. Згідно одного символу із множини можливих {‘F’,’R’,’C’}, що повертає запит опонент (комп’ютер) приймає рішення, щодо тієї ситуації, що він перебуває. Наприклад з бази даних повернуто «F», то опонент (комп’ютер) скидає карти, «R» опонент підвищує ставку або «C» опонент зрівнює ставку. Якщо опонент не скинув карти, то виконуються всі подальші дії, які описані.

3. Передамо руку («CA DJ») та флоп («DA C2 ST») в метод для складання комбінації.

3.1 В методі першим кроком застосовується операція конкатенації [15] стрічок.

3.2 В масив індексів записуються індекси рангів карт з новоутвореної стрічки.

3.3 До масиву індексів застосовується метод Хоара [16], який сортує карти по спаданню їх рангів карт.

3.4 По масиву індексів знаходимо ранг карти в масиві рангів карт.

3.5 Створюємо по масиву індексів відсортовану стрічку, тобто «DA CA DJ ST C2», рисунок 3.5.

Рисунок 3.5 – Відсортована стрічка карт «DA CA DJ ST C2»

3.6 До стрічки утвореної в підпункті 3.5 застосовується метод, який підрахує кількість входжень рангу кожної карти в стрічці, результатом цієї операції буде стрічка «2 2 1 1 1», рисунок 3.6.

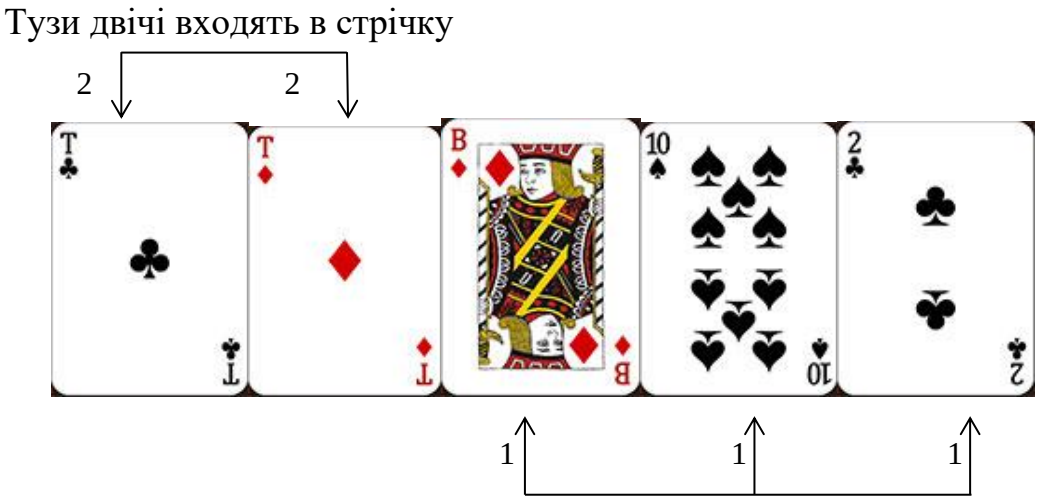


Рисунок 3.6 – Застосування операції «Кількість входжень» до стрічки «DA CA DJ ST C2»

3.7 Дві двійки в підпункті 3.6 означають, що гравець має пару тузів.

3.8 Для терна та для рівера виконуються аналогічні операції з підпункту 1.1 по підпункту 3.8.

4. Якщо комбінацію не вдалося скласти, то виконується операція селекції до бази даних [17] в якій буде міститися таблиця 2.1, в запиті буде повертатися вся стрічка, що відповідає даній комбінації, що є в гравця.

5. Для приведеного прикладу повертається стрічка в такому вигляді: «$Outs = 3;\ \ Flop\_to\_Tern = 5{,}84;\ \ Tern\_to\_River = 8{,}2;\ \ Flop\_River = 3{,}91;$ 2_Pair», «2_Pair», ця стрічка показує комбінацію, яку можна скласти, якщо прийде одна із 3 карт, що залишилися в колоді. По інформації, що отримана з бази даних підраховуються доцільність ставки по формулам 2.5 та 2.6.

5.1 Одси (шанси банка) для кожного етапу гри отримуються на кроці 2, з таблиці 2.1 відповідно до стадії гри, наприклад стадія гри «Флоп», то обирається колонка «$Flop \rightarrow Tern$».

5.2 Якщо банк = 100, а ставка = 20, то пот-одси (потенційні шанси банку) будуть дорівнювати 20%.

5.3 На стадії гри «$Flop \rightarrow Tern$» одси будуть дорівнювати 5,84, 5,84 < 20, ставка не виправдана.

6. Перевірка сили комбінації кожного із гравців.

7. Якщо комбінація гравця краща, то виграв гравець, інакше опонент.

8. Для кожної із стадіїв гри повторюються кроки з 1 по 5.

З аналізу роботи алгоритму видно, що алгоритм на кожній стадії гри виконує ту послідовність дій, які виконує професійний гравець сидячи за гральним столом, на кожному етапі гри.

3.3 Розробка алгоритму прийняття рішення в «Техаському холдемі»

Алгоритм прийняття рішення [18] в «Техаському холдемі» буде реалізований в окрему класі, який буде містити всі необхідні методи для оціночної функції. Даний клас буде викликатися не тільки для опонента (комп’ютера), а також буде викликатися для гравця. За допомогою цього класу гравцеві на кожному етапі гри будуть підраховуватися:

1. Шанси банку;
2. Потенційні шанси банку;
3. Шанси на покращення;
4. Аути;
5. Математичне сподівання;
6. Комбінація.

Всі інші операції для гравця в даному класі не будуть виконуватися, для цього при виклику даного класу буде передаватися булівська змінна, яка при виклику методу буде кожен раз перевірятися, якщо «істина», то це опонент (комп'ютер), інакше ця змінна буде мати значення «хибність», що в свою чергу заборонить виконання віток умов.

Алгоритми, які застосовує опонент (комп'ютер) на кожному етапі гри:

1. На префлопі опонент (комп'ютер) звертається до бази даних MS SQL, в якій містяться комбінації двох карт, і за допомогою мови LINQ проводиться запит до цієї бази даних, в якій міститься дві таблиці «$Big\ Blind$», «$Small\ Blind$» (в залежності від того, який блайнд ставить опонент проводиться запит до тієї таблиці). Запит повертає char-символ {'R','C','F'}, який буде містити оцінку тієї ситуації, що склалася в опонента. Якщо було повернуто 'F', то опонент (комп'ютер) скидує карти, і банк переходить до гравця, якщо повернуто 'C', то опонент робить колл (зрівнює ставку), 'R' опонент робить рейз (підвищує ставку). В разі повернення 'C' або 'R' після зрівняння ставок виконується наступний крок.

2. Визначити кількість аутів (карти, які можуть покращити руку) в колоді. Щоб визначити кількість аутів потрібно порахувати найкращу комбінацію для опонента (комп'ютера). На флопі найкраща комбінація це його рука та флоп, тому що найкраща комбінація в «Техаському холдемі» складається з 5 карт, а так як роздається всього 5 карт (2 карти рука, 3 карти флоп), то вважається, що це його найкраща комбінація. На терні і рівері будуть також визначатися найкращі 5 карт. По найкращим п'ятьом картам

визначається комбінація опонента (комп’ютера), яка передається в запиті до бази даних в якій містяться всі комбінації, наперед визначені аути для кожної із комбінації та ймовірність складання даної комбінації на наступному етапі гри.

3. Також з бази даних проводиться операція селекції по «$Odds$» шансам банку, які відповідають кожній із комбінацій у відсотках.

4. Розраховуються шанси на покращення комбінації за формулою 2.4.

5. Розраховується сила руки, яку має опонент (комп’ютер). Підрахування сили руки проводиться на базі тієї комбінації, що має опонент (комп’ютер). Сила руки вимірюється в числовому еквіваленті, відповідно чим краща комбінація тим більша сили руки. В залежності від комбінації, яку має опонент буде додаватися коефіцієнт, для кожної комбінації він буде різним. Коефіцієнти наведені нижче:

5.1 Одна пара – 1300 + найкращі 3 карти + індекси пари.

5.2 Дві пари – 2600 + кікер + індекси двох пар.

5.3 Сет – 3900 + найкращі 2 карти + індекси трьох карт сета.

5.4 Стріт – 5200 + індекси кожної із карт стріта.

5.5 Флеш – 6500 + індекси кожної із карт флеша.

5.6 Фулл-хаус – 7800 + індекси кожної карти із пари + індекси кожної карти із сета;

5.7 Каре – 9100 + індекси кожної із 4 карт + кікер.

5.8 Стріт-флеш – 10400 + індекс кожної із карт стріта.

5.9 Флеш-рояль – 117000.

5.10 Кікер – 10 * індекс кожної карти, яка є найкращої для гравця.

6. Підраховується доцільність колу. В даному методі підраховується можливий виграш та можливий програш, для цього використовуються формули: 2.5 – 2.7.

7. Розраховується математичне сподівання за формулою 2.10.

8. Якщо колл є доцільнім для даної ігрової ситуації, переходимо до наступних кроків.

9. Розраховується актуальність ставки, для цього використовується формула 2.11.

10. Підраховуються потенційні шанси банку, використовується формула 2.12.

11. Якщо сума шансів на покращення, математичного сподівання більша ніж потенційні шанси банку і доцільно робити колл (якщо опонент поставив ставку), то зрівнюється ставка.

12. Якщо сума шансів на покращення, математичного сподівання більша ніж потенційні шанси банку і доцільно робити колл, в той час, коли опонент не ставив ставку, то ставиться ставка.

13. Якщо сума шансів на покращення, математичного сподівання набагато більша ніж потенційні шанси банку і доцільно робити колл, і якщо опонент поставив ставку, то ставка підвищується (рейз).

14. Якщо сума шансів на покращення, математичного сподівання менша ніж потенційні шанси банку і опонент не ставив ставку, то пропускаємо хід (чек).

15. Якщо сума шансів на покращення, математичного сподівання менша ніж потенційні шанси банку і опонент не ставив ставку, то гравець скидає карти (фолд).

Граф-схема алгоритму прийняття рішення в «Техаському холдемі» наведена на рисунку 3.7.

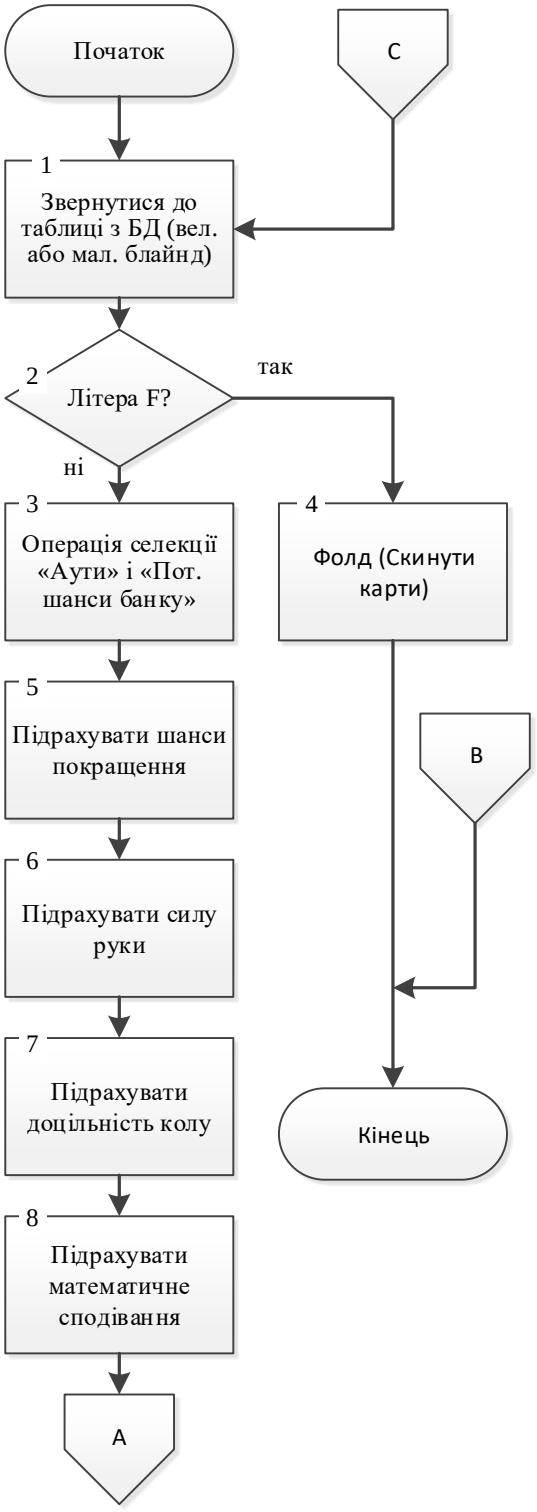


Рисунок 3.7 – Граф -схема алгоритму прийняття рішення в «Техаському холдемі»

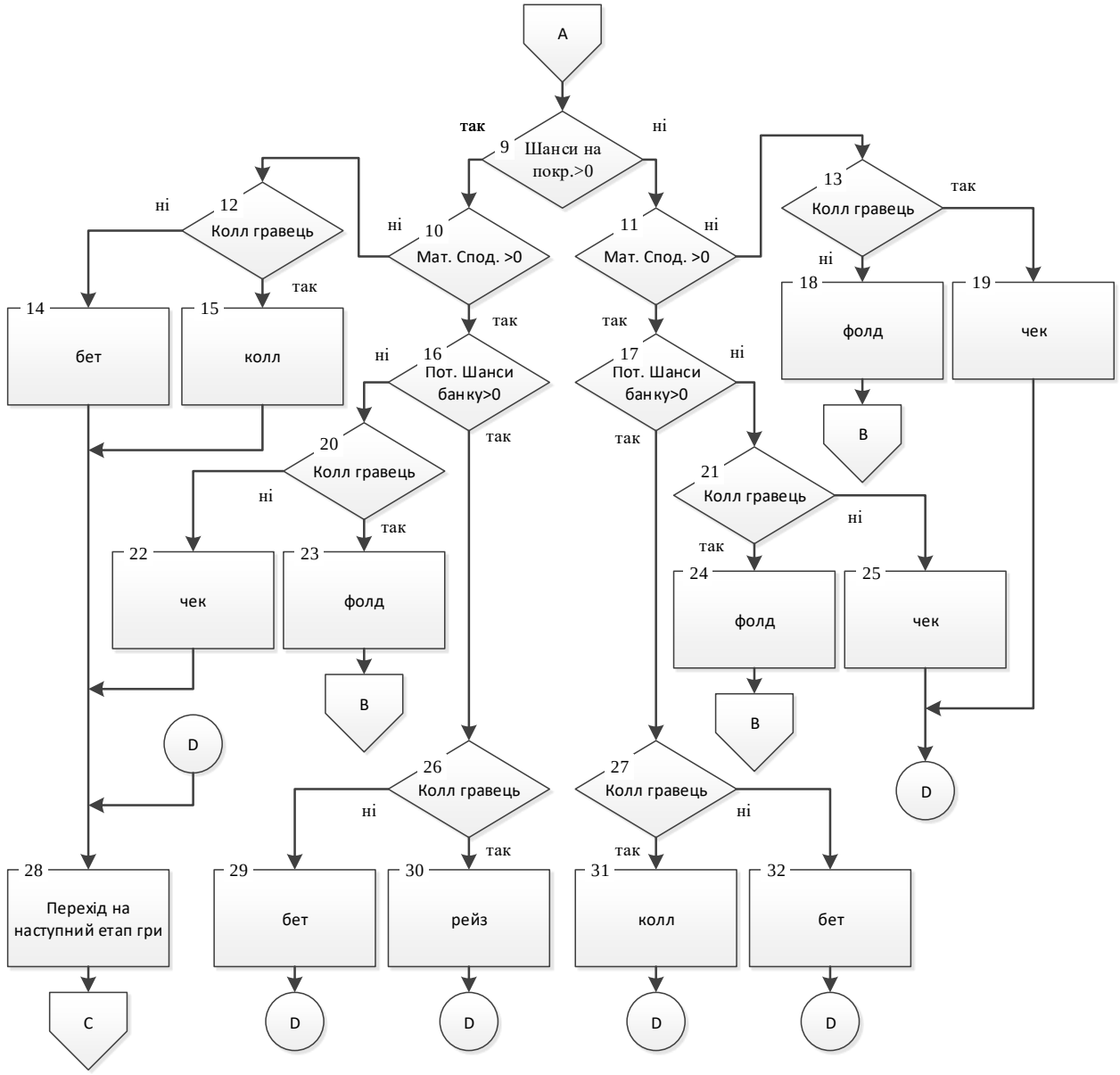


Рисунок 3.7 (продовження)

Розроблений алгоритм прийняття рішень в «Техаському холдемі» наділяє опонента (комп'ютер) інтелектом, достатнім для того, щоб скласти достойну конкуренцію гравцеві.

3.4 Розробка програмного забезпечення гри «Техаський холдем»

Згідно вище наведеним опишемо головний клас гри, який буде містити інтелектуальні методи. Клас, який буде містити всі методи, він складається з 8 методів. Наведемо назви методів та дії, які вони виконують:

1. *1. $Actual\ rate$ – головний метод, який буде мати специфікатор доступу public (відкритий). Цей метод буде викликатися на кожній стадії гри, який в свою чергу буде викликати кожен раз всі нижче наведені методи. На основі тих даних, які зможе даний метод зібрати від інших методів буде прийматися остаточне рішення.*
2. $String\ call\ preflop$ – метод, який підраховує доцільність ставки на префлопі, для цього стрічка з рукою опонента буде сортуватися за спаданням (від більшого до меншого) і, якщо карти різномасті буде додаватися маленька латинська буква «s», інакше « » (пропуск), так як поле в базі даних складається з трьох символів. Підготовлена стрічка використовується в LINQ запиті, який проводиться до бази даних, в якій містяться таблички див. рисунок 2.8 та 2.9. З бази даних повертається символ з множини можливих {‘C’,’F’,’R’}, якщо ‘C’, то колл (зрівняти ставку), ’F’ фолд (скинути карти), ’R’ рейз (підвищити ставку).
3. $Chances\ for\ improvment\ (S)$ – метод підраховує шанси на покращення, згідно формули 2.4.

Кількість карт в колоді визначається що разу, як створюється екземпляр об’єкту класу. Кожен раз як створюється екземпляр об’єкту класу в конструктор передається один параметр «$table$», який визначає стадію гри флоп, терн чи рівер. По цьому параметру визначається кількість карт в колоді, що залишилися без врахування карт опонента. Наприклад:

3.1 $Table\ =\ 1$, то кількість карт в колоді = 47, тому що це флоп.

3.2 $Table\ =\ 2$, то кількість карт в колоді = 46, тому що це терн.

3.3 $Table\ =\ 3$, то кількість карт в колоді = 45, тому що це рівер.

4. $Call\ (C)$ – метод підраховує доцільність колу, для цього рахує можливий виграш та програш по формулах 2.5, 2.6 та 2.7:

5. $Expected\ value\ (E)$ – метод підраховує математичне сподівання за формулою 2.10.
6. $Strength\ of\ hand\ (H)$ – метод підраховує силу руки по відношенню до інших комбінацій. Даний метод проводить розрахунки на основі вже існуючої комбінації. Це дасть змогу оцінити шанси на виграш. Даний метод запускається в тому випадку, якщо є якась комбінація не кікер. Для кікера запускається метод $Strength\ of\ hand\ kiker$.
7. $Strength\ of\ hand\ kiker\ (H)$ – метод дозволяє визначити силу руки по кікеру.
8. $The\ whole\ rate$ – комп'ютер обчислює розмір ставки згідно математичних формул, вони не завжди кратні десяти, тому, щоб зробити ставки комп'ютера кратними десяти було введено даний метод.

3.5 Висновок до розділу 3

В даному розділі розроблено оціночну функцію, яка в свою чергу побудована на основі математичної моделі гри, яка наведена в розділі 2. Також розроблено алгоритм інформаційної технології гри в «Техаському холдемі», розроблений алгоритм передбачає моделювання роботи справжнього опонента гравця. На основі вищенаведеної інформації розроблено програмний продукт на об'єктно-орієнтованій мові C#.

4 ТЕСТУВАННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ РОБОТИ ПРОГРАМИ МОДЕЛЮВАННЯ ІГРОВОЇ СИТУАЦІЇ «ТЕХАСЬКИЙ ХОЛДЕМ»

4.1 Тестування програми моделювання ігрової ситуації «Техаський холдем»

В ході виконання магістерської роботи було проаналізовано основні підходи щодо моделювання та створення гри покер «Техаський холдем».

Гра починається з виведення на екран дисплея стартового вікна гри (рисунок 4.1), в даному вікні гравець може запустити гру, продивитися турнірну таблицю або закрити програму.

Рисунок 4.1 – Стартове вікно гри

На рисунку 4.2 подана турнірна таблиця результатів гри в покер «Техаський холдем», турнірна таблиця також передбачає виконання чотирьох запитів:

1. «Знайти гравця по введеному імені» в таблиці;
2. «Впорядкувати по стеку опонента» в таблиці;
3. «Впорядкувати по стеку гравця» в таблиці;
4. «Впорядкувати по імені гравця» в таблиці.

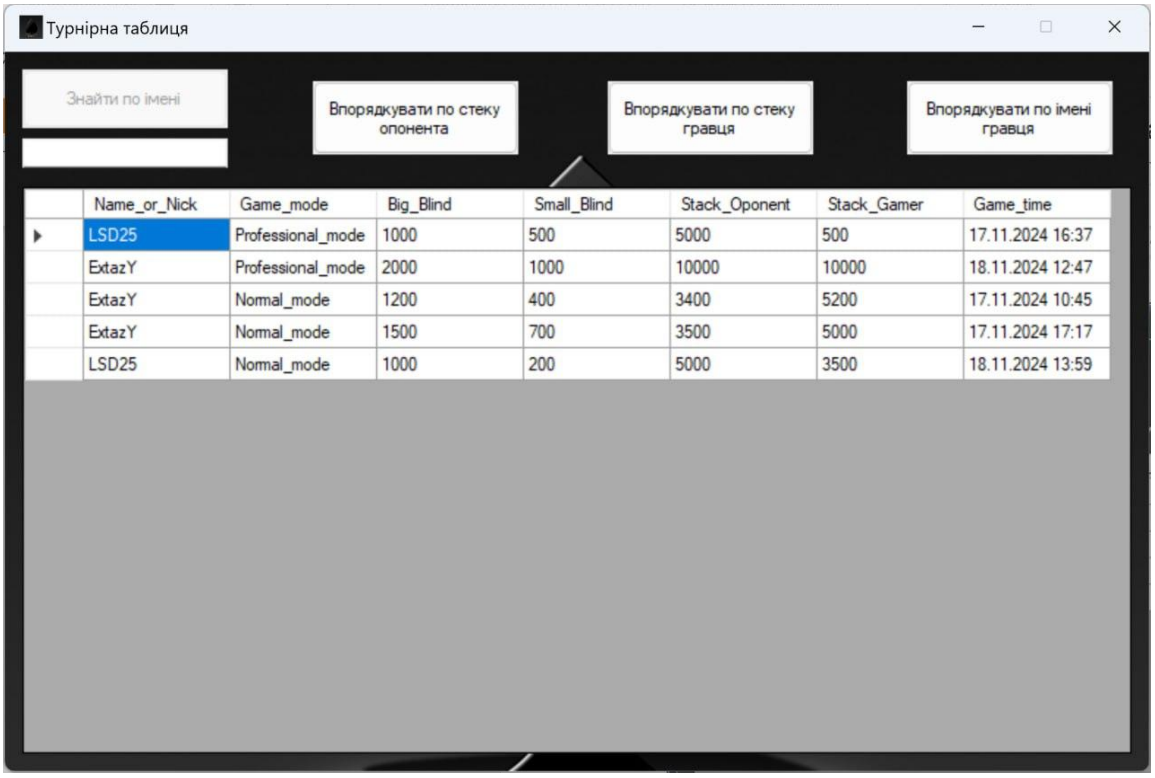


Рисунок 4.2 – Турнірна таблиця результатів гри в «Техаський холдем»

В турнірній таблиці відображається ім’я або нік гравця, режим гри (звичайний чи професійний), великий блайнд, малий блайнд, стек опонента, стек гравця та час закінчення гри, всі ці параметри користувач задає, коли завантажує гру, також форму, де вводяться параметри гри користувач може завантажити в будь-який момент гри через меню головного вікна гри.

На рисунку 4.3 поданий результат запиту «Знайти гравця по введеному імені», на рисунку 4.4 результат запиту «Впорядкувати по стеку опонента», на рисунку 4.5 результат запиту «Впорядкувати по стеку гравця», на рисунку 4.6 результат запиту «Впорядкувати по імені гравця».

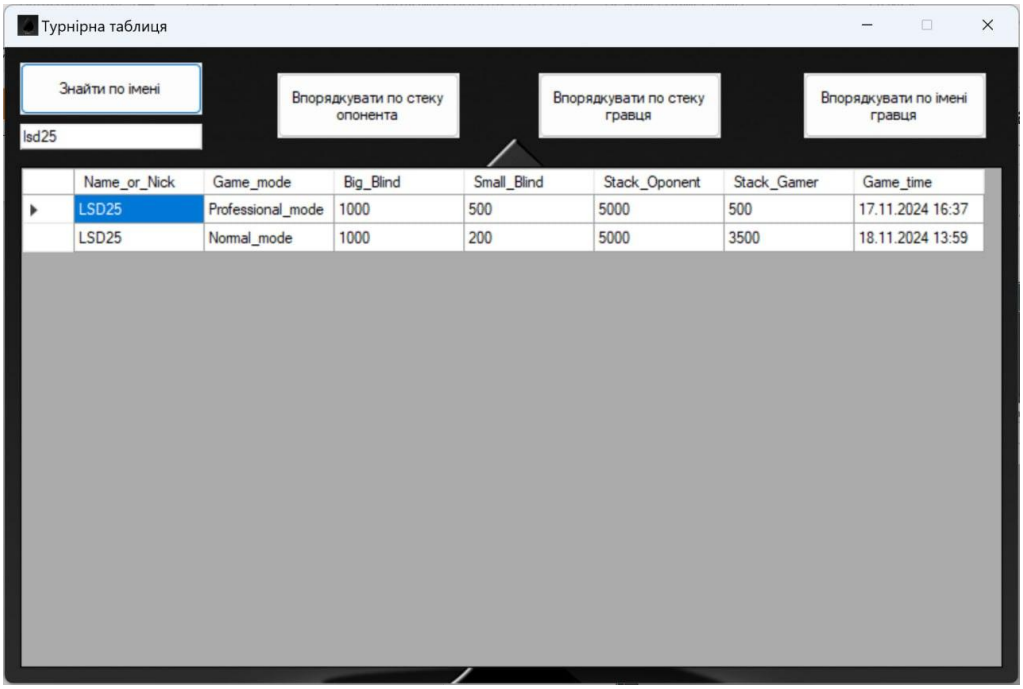


Рисунок 4.3 – Результат запиту «Знайти гравця по введеному імені»

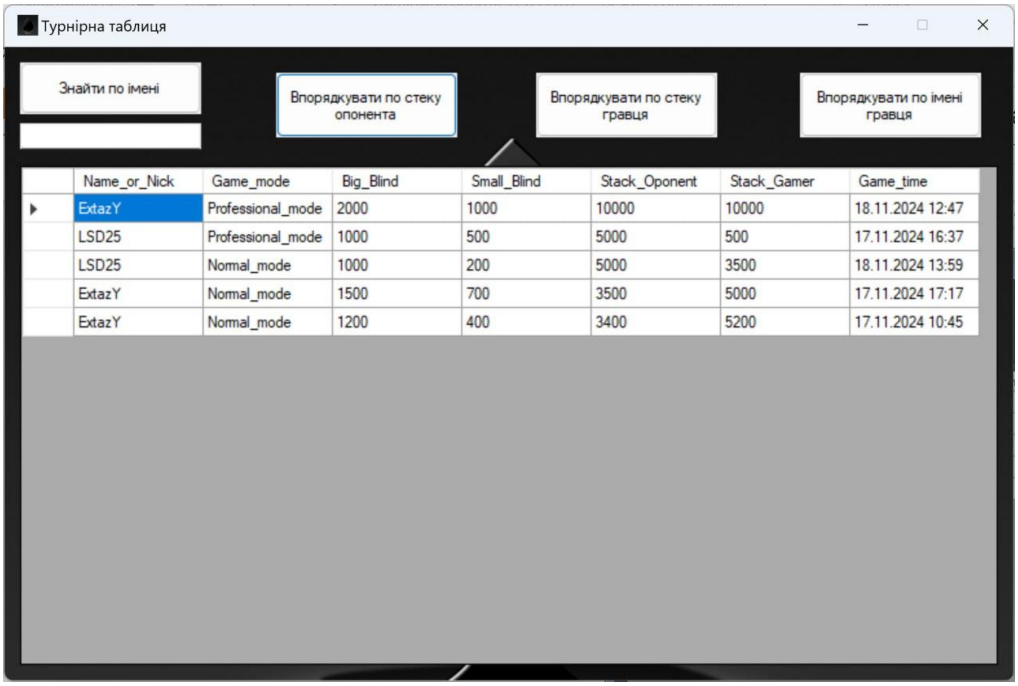
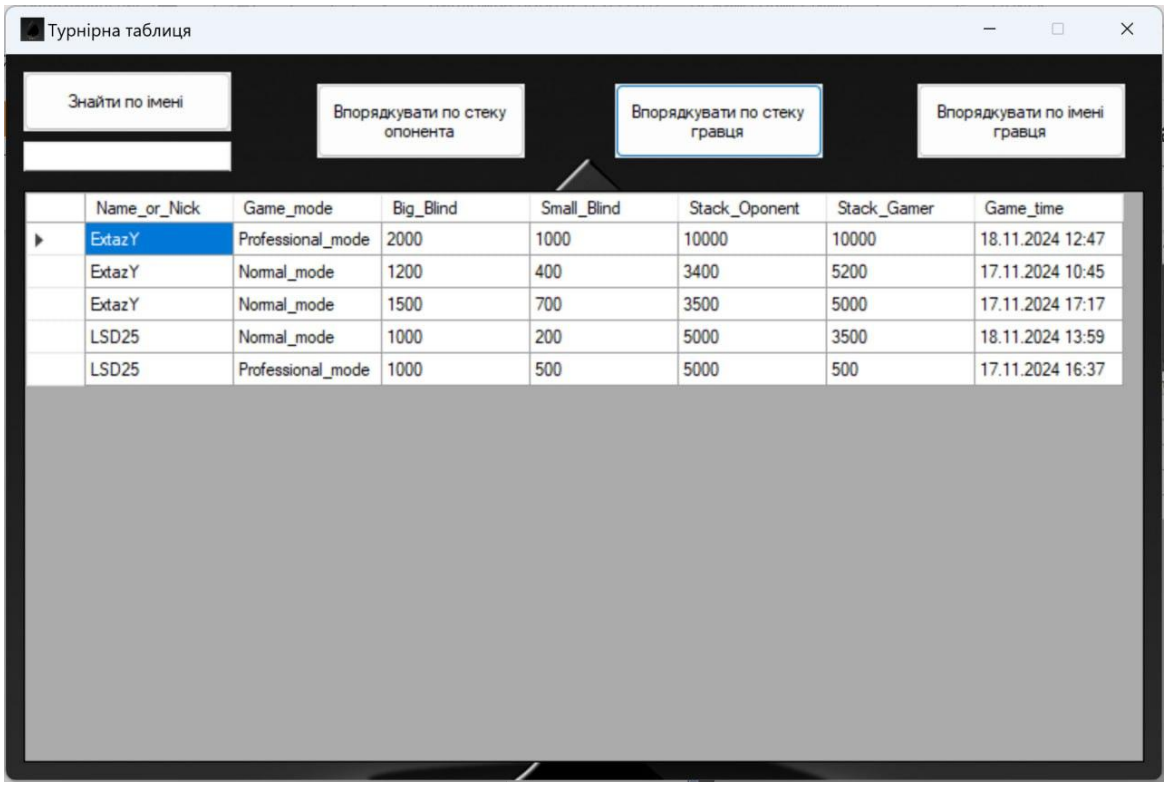
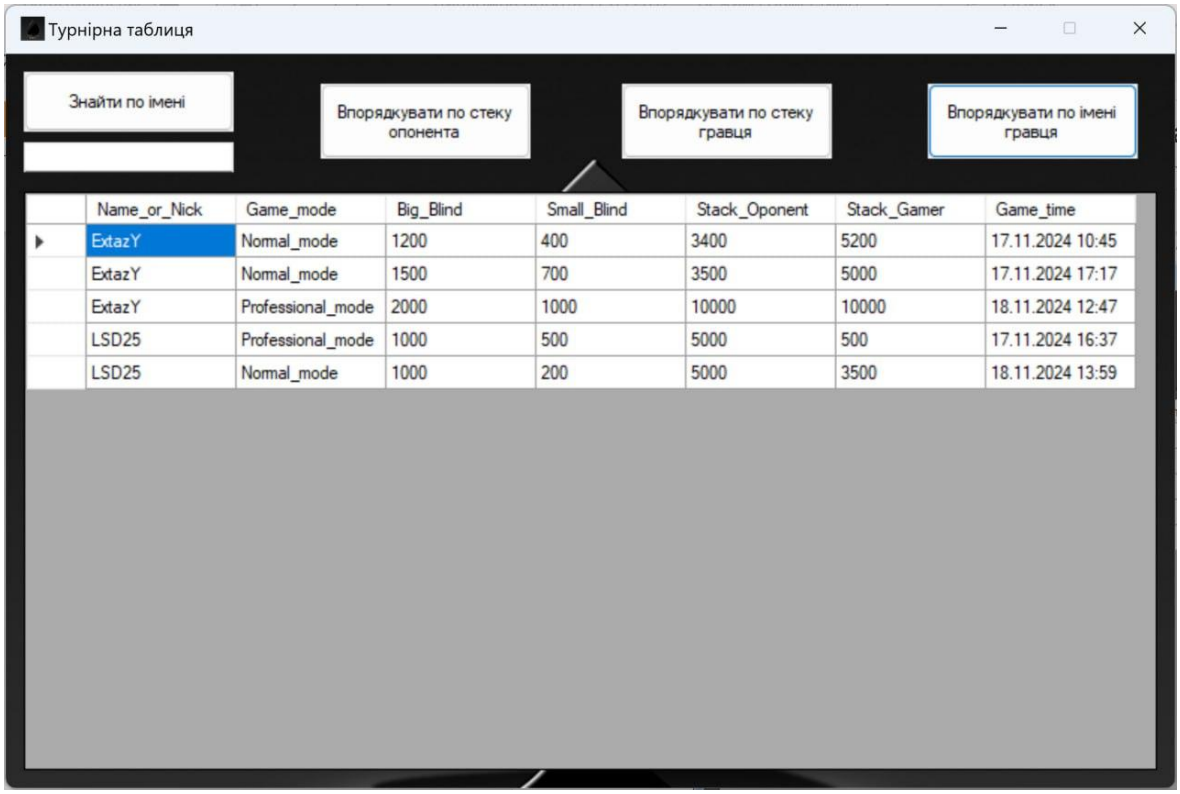


Рисунок 4.4 – Результат запиту «Впорядкувати по стеку опонента»



Name_or_Nick	Game_mode	Big_Blind	Small_Blind	Stack_Oponent	Stack_Gamer	Game_time
ExtazY	Professional_mode	2000	1000	10000	10000	18.11.2024 12:47
ExtazY	Normal_mode	1200	400	3400	5200	17.11.2024 10:45
ExtazY	Normal_mode	1500	700	3500	5000	17.11.2024 17:17
LSD25	Normal_mode	1000	200	5000	3500	18.11.2024 13:59
LSD25	Professional_mode	1000	500	5000	500	17.11.2024 16:37

Рисунок 4.5 – Результат запиту «Впорядкувати по стеку гравця»



Name_or_Nick	Game_mode	Big_Blind	Small_Blind	Stack_Oponent	Stack_Gamer	Game_time
ExtazY	Normal_mode	1200	400	3400	5200	17.11.2024 10:45
ExtazY	Normal_mode	1500	700	3500	5000	17.11.2024 17:17
ExtazY	Professional_mode	2000	1000	10000	10000	18.11.2024 12:47
LSD25	Professional_mode	1000	500	5000	500	17.11.2024 16:37
LSD25	Normal_mode	1000	200	5000	3500	18.11.2024 13:59

Рисунок 4.6 – Результат запиту «Впорядкувати по імені гравця»

Коли користувач натисне на кнопку «Запустити гру» див. 4.1 йому будуть запропоновані головні параметри гри, в цих параматрах користувач зможе обрати режим гри (Професійний чи звичайний) задати розміри блайднів та розміри стеку, рисунок 4.7.

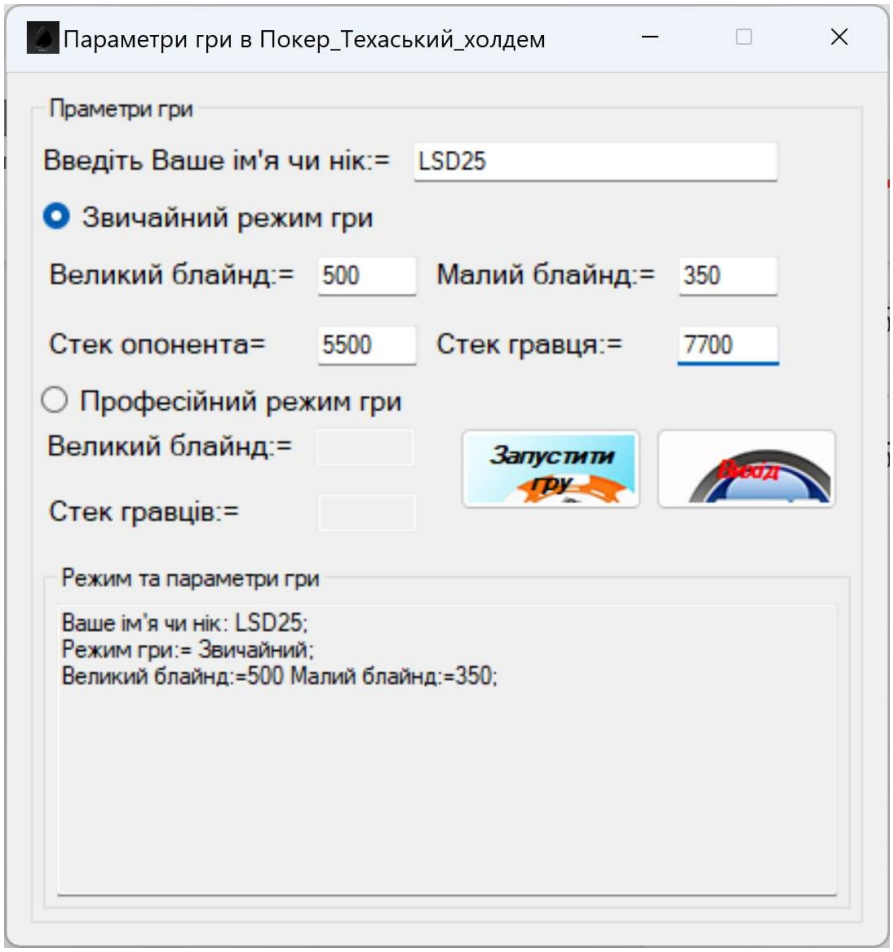


Рисунок 4.7 – Головні параметри гри в «Техаському холдемі»

В звичайному режимі гри користувачу буде відображатися стрічка в якій будуть містити головні параметри ситуації, яку має гравець. В цій стрічці будуть міститися «Шанси банку», «Потенційні шанси банку», «Шанси на покращення», «Аути», «Математичне сподівання» та «Комбінацію», яку гравець має на момент гри. Ця стрічка значно спрощує гру, дозволяючи більш доцільно визначити ситуацію в якій знаходиться гравець. В професійному режимі гри даної стрічки немає.

Професійний режим також передбачає те, що малий блайнд встановлюється згідно правил покеру (половина від великого блайнду), тому користувачу запропоновано для вводу тільки великий блайнд, також стек обох гравців встановлюється рівним як для опонента так і для гравця. На рисунку 4.8 поданий рисунок головної форми гри покер «Техаський холдем».

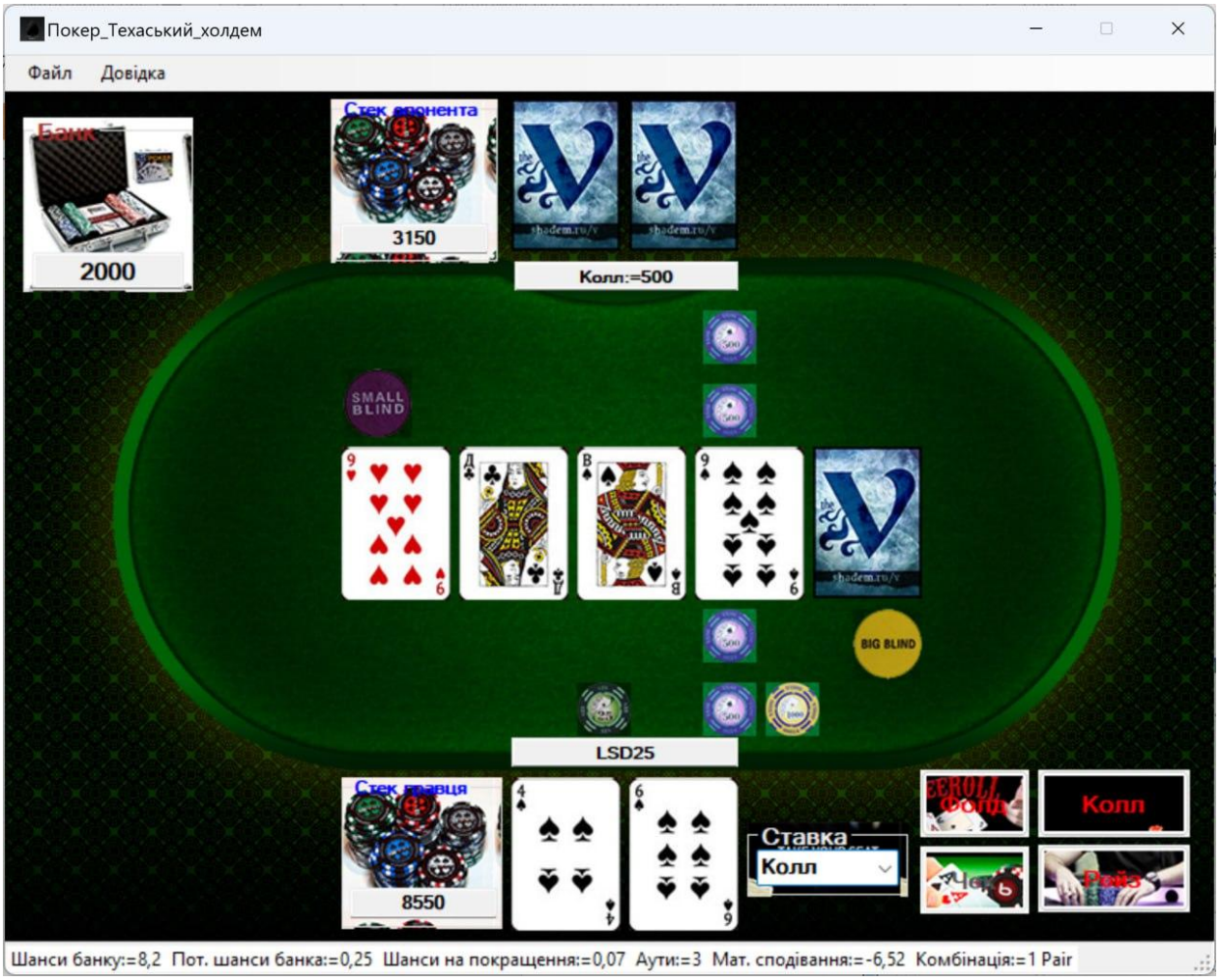


Рисунок 4.8 – Головна форма гри «Техаський холдем»

Зверху форми розташоване головне меню гри, дане меню «Файл» надає можливість закрити програму, відключити стрічку в якій містяться головні параметри ігрової ситуації, та завантажити форму з параметрами гри див. рисунок 4.7. Меню «Довідка» наводить правила гри програми та як користуватися грою.

В лівому верхньому кутку відображається банк гри, він включає всі блайнди та ставки, які гравці зробити на протязі гри. Правіше від банку знаходиться стек опонента та карти опонента. Під картами опонента розміщується поле, в якому відображається слово опонента, нижче даного поля розміщуються фішки опонента. Фішки під час гри можуть розмінюватися в залежності від стеку опонента, тобто під час гри можуть з'являтися номінали фішок, яких раніше не було. Наприклад, якщо в опонента стек дорівнює 9050, то його можна представити у вигляді номіналів фішок «1000» та «25», рисунок 4.9. Опонент ставить малий блайнд у розмірі 25, потім робить колл у розмірі 220, його стек стане рівним 8805, то його фішки можна представити за допомогою «5», «100», «500» та «1000», рисунок 4.10.

Рисунок 4.9 – Стек опонента в розмірі «9050»

Рисунок 4.10 – Стек опонента в розмірі «8805»

По середині столу розташовані карти, які роздаються на кожній стадії гри. Для карт, які закриті відображається так звана «Рубашка» карт, рисунок 4.11.

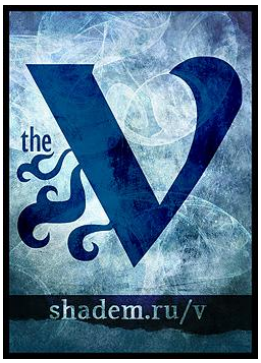


Рисунок 4.11 – «Рубашка» карт

Нижче карт стола розташовані фішки стеку гравця, під фішками розта шоване поле «Ім’я гравця», то ім’я, яке користувач вводить в формі «Параметри гри в покер Техаський холдем» див. рисунок 4.7. Під полем «Ім’я гравця» розташована «Рука» гравця, правіше від руки гравця розташований стек гравця. Лівіше від руки гравця знаходиться поле в якому можна обирати розмір ставки (розмір ставки кратний 10), правіше розташовані кнопки, які користувач використовує для гри в покер «Техаський холдем». Для кращої навігації окрім кнопок, користувач може користуватися контекстним меню, яке викликається натисканням правої кнопки мишки, на рисунку 4.12 подане контекстне меню гри покер «Техаський холдем».

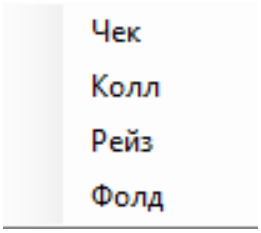


Рисунок 4.12 – Контексне меню гри покер «Техаський холдем»

Довідка до розробленої гри в покер «Техаський холдем» написана на трьох мовах – англійській, російській та українській, що дасть змогу її прочитати більшій кількості людей, на рисунку 4.13 подана довідка українською мовою.

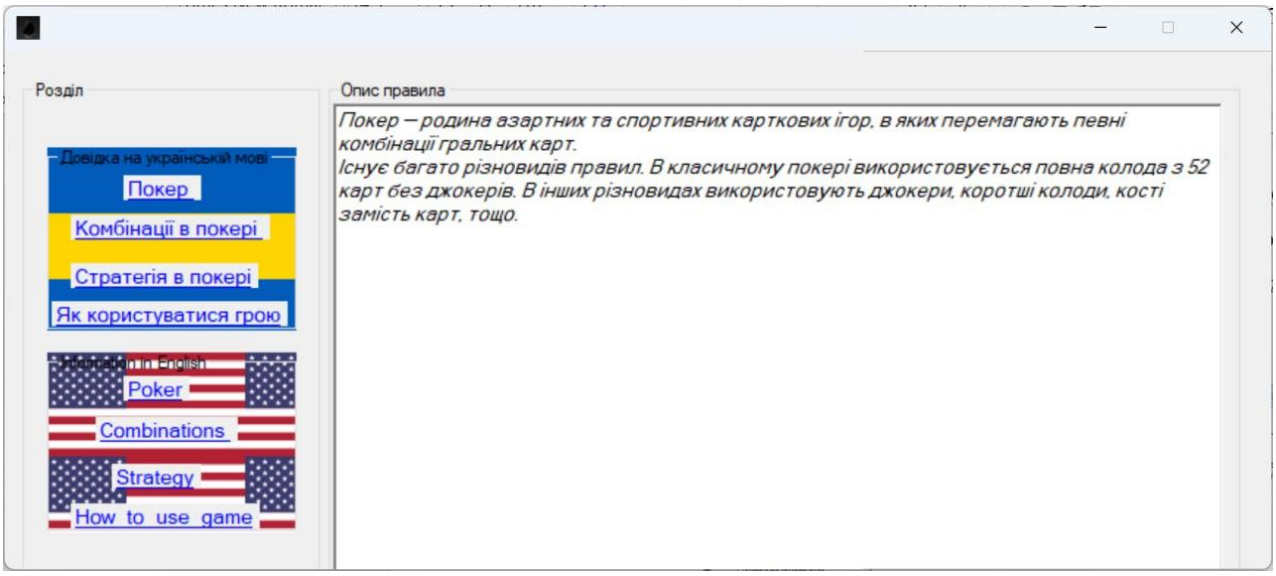


Рисунок 4.13 – Довідка українською мовою

4.2 Аналіз результатів роботи програми моделювання ігрової ситуації «техаський холдем»

Розглянемо процес гри в покер «Техаський холдем», коли опонент (комп’ютер) ходить першим і ставить малий блайнд, рисунок 4.14.

В параметрах гри було обрано малий блайнд в розмірі 350, великий блайнд 500, стек опонента задано 7500, так як опонент поставив малий блайнд в розмірі 350, то його стек став рівним 7150 (7500-350=7150), після того як опонент поставив ставку є три варіанти гри для гравця:

1. Поставити великий блайнд;
2. Зробити рейз (підвищити ставку);
3. Скинути карти (фолд).

Для продовження гри оберемо перший варіант, тобто поставимо великий блайд. Після того як поставлені обов’язкові ставки (блайнди), роздаються карти флопа (3 карти), на рисунку 4.15 подані карти флопа.

Рисунок 4.14 – Процес гри в покер «Техаський холдем»

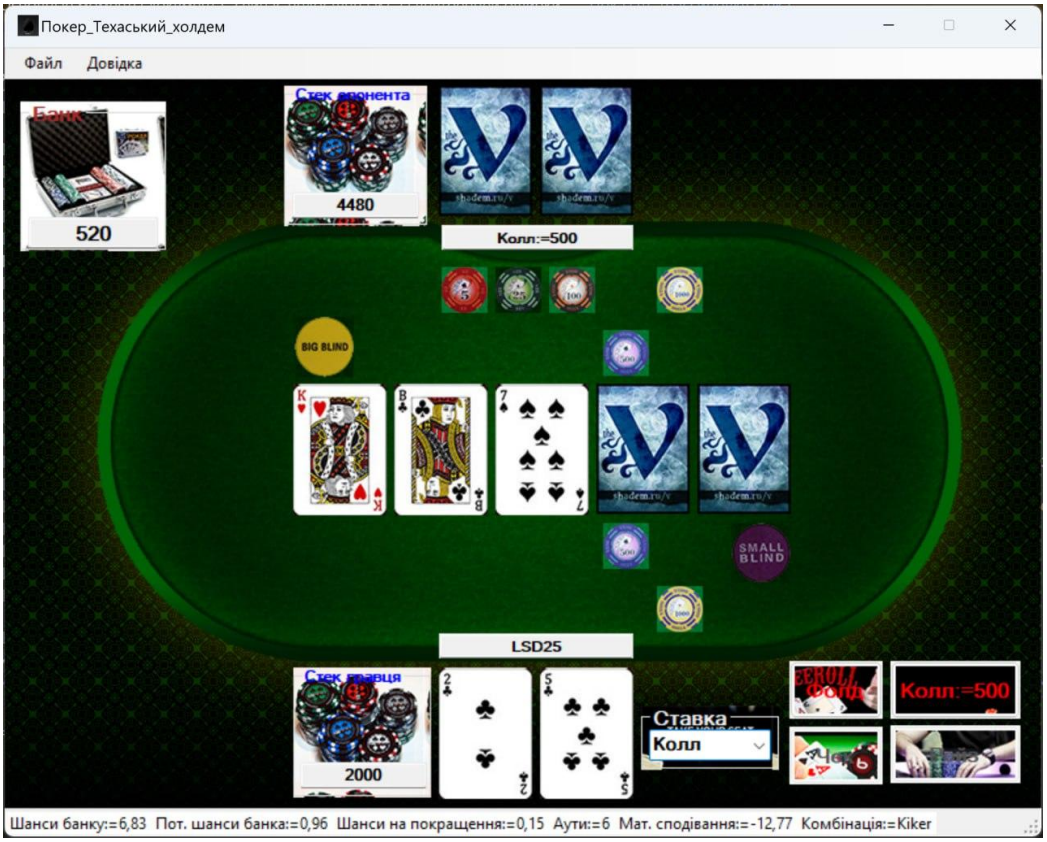


Рисунок 4.15 – Карти флопа

Так як гравець не поставив ставку (колл) і опонент (комп’ютер) немає комбінації, опонент приймає рішення пропустити хід «чек», поставимо також «чек», після цього роздадуться карти терна, рисунок 4.16.

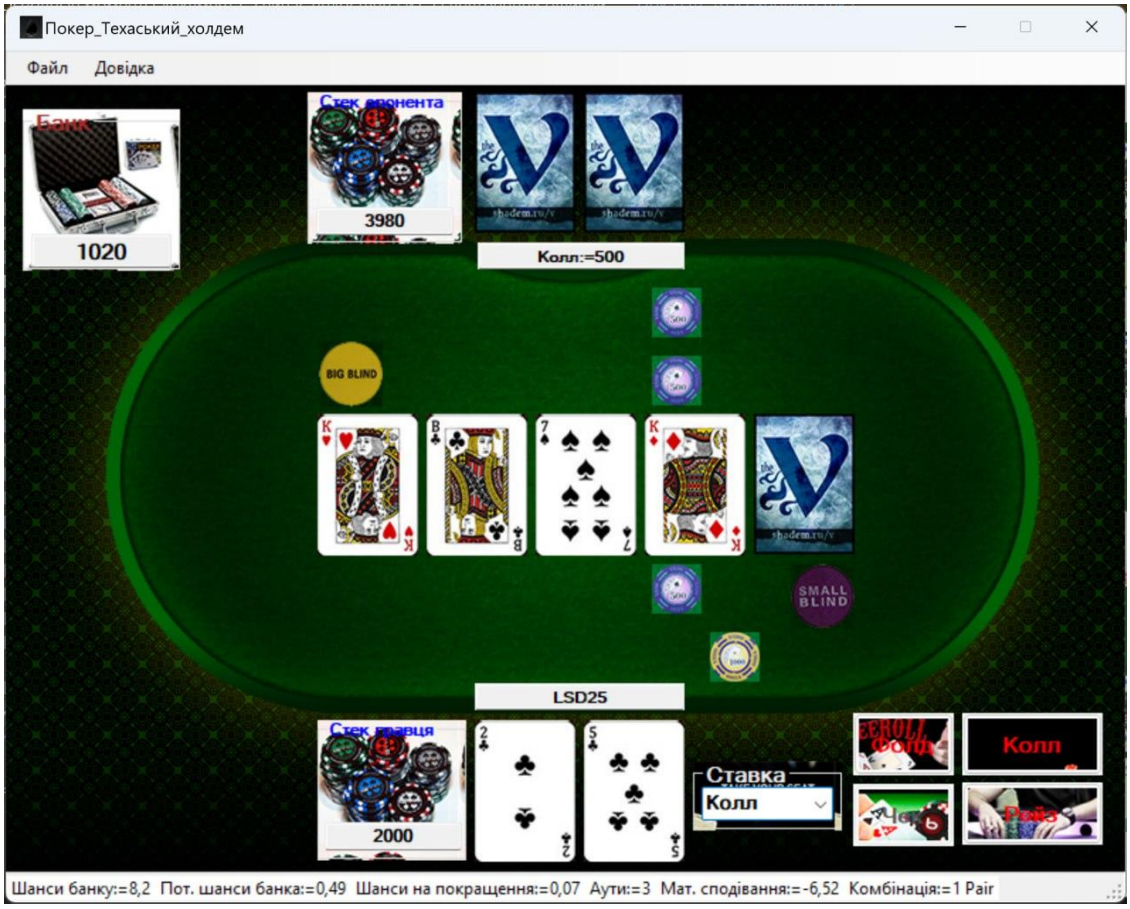


Рисунок 4.16 – Карти терна

На терн прийшла «десятка чирви» вона не покращує комбінацію руки опонента (комп’ютера), тому опонент приймає рішення пропустити хід «чек», а для гравця є дуже високі шанси на виграш, а саме:

1. Шанси банку = 4.75;
2. Потенційні шанси банку = 0, згідно формули 2.2.9 в чисельнику розміщується ставка, яку поставив опонент (комп’ютер), так як опонент не ставив ставку, то відповідно чисельник цієї формули рівний нулю.

3. Шанси на покращення = 0.21;

4. Аути = 8;

5. Математичне сподівання того, що прийде один із аутів = 17.39;

6. Комбінація = «2 Straight dro».

В опонента достатньо добрі шанси, щоб продовжувати гру він має двухстороннє стріт-дро, іншими словами можна зібрати стріт, використовуючи не 4 карти як при звичайному, а 8, що в свою чергу збільшує шанси на виграш опонента. Але стріт-дро та двухсторонній стріт-дро не готові комбінації, тому що стріт-дро складається з чотирьох карт, а комбінація в «Техаському холдемі» складаються з п’яти карт.

Проаналізувавши ігрову ситуацію для гравця, можна прийти до висновку, що гравець має достатньо високі шанси на складання стріта, в гравця є двухсторонній стріт-дро, при цьому якщо випаде один із чотирьох валетів, то гравець буде мати досить сильну руку з якою можна грати майже при будь-яких ставках.

Поставимо ставку в розмірі 1000 (колл 1000), опонент (комп’ютер) скоріше за все скине карти, тому що опонент немає комбінації з якою можна було б грати при даній ставці, на рисунку 4.17 показаний результат даного ходу. Як і передбачалося опонент скинув карти, тому що можливий програш за даної ставки набагато більший ніж можливий виграш, також математичне сподівання має від’ємне значення, саме через це опонент скидає карти, тобто робить «Фолд».

Як і передбачалося опонент (комп’ютер) порахував, що він немає готової комбінації, тому можливий програш при даній комбінації для опонента (комп’ютера) був набагато більший ніж виграш (див. формули 2.2.2 – 2.2.6), внаслідок цього математичне сподівання приймає від’ємне значення, а булівська змінна «C» прийме значення false, саме через це опонент (комп’ютер) скидає карти.

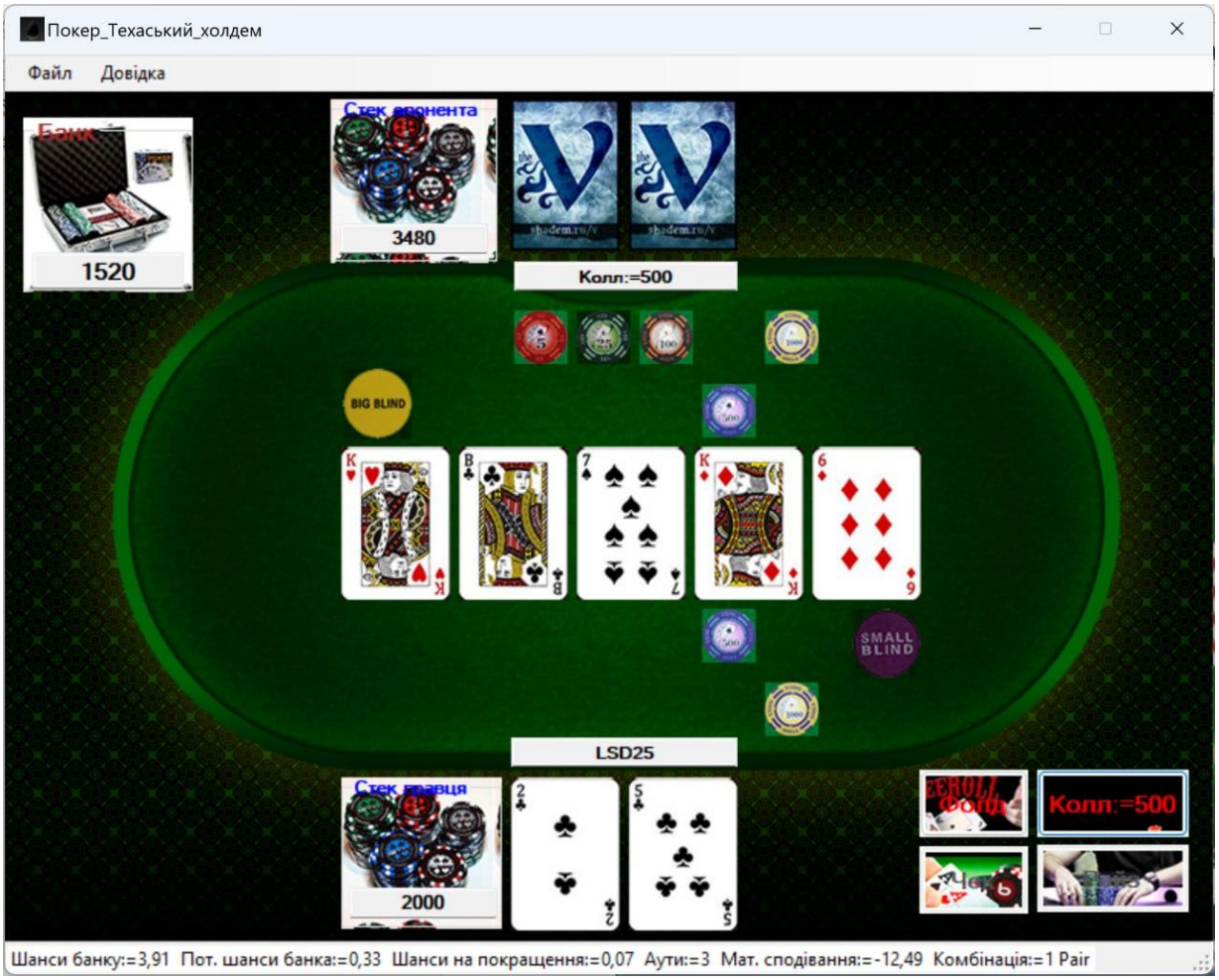


Рисунок 4.17 – Результат колу гравця

Проаналізувавши вище наведені рисунки (див. рисунки 4.14 – 4.17) можна прийти до висновку, що алгоритм приймає вірне рішення щодо ігрової ситуації, в якій він знаходився.

Порівняємо функціональні можливості розробленої програми на найкращого аналога (програми Pokerstars), які зведено у табл. 4.1

Таблиця 4.1 – Порівняння функціональних можливостей розробленої програми на програми-аналога

№	Функції роботи програми	Програма-аналог (Pokerstars),	Розроблена програма
1.	Гра безкоштовна	-	+
2.	Відсутня реклама	-	+

Таблиця 4.1 (продовження)

№	Функції роботи програми	Програма-аналог (Pokerstars),	Розроблена програма
3.	Професійний режим гри	–	+
4.	Розрахунок шансів банку	–	+
5.	Розрахунок потенційних шансів банку	–	+
6.	Розрахунок шансів на покращення	–	+
7.	Розрахунок аутів	–	+
8.	Розрахунок мат. сподівання	–	+
9.	Розрахунок поточної комбінації	–	+
10.	Чат	+	-
11.	Мобільна версія	+	-
12.	Роздача карт	+	+
13.	Тасування карт (методом Хоара)	-	+
14.	Роздача фішок (жадібний алгоритм)	-	+

Із табл. 4.1 видно, що програма-аналог має 3 функції роботи, а розроблена програма має 14 функцій роботи. Це означає, що мета магістерської кваліфікаційної роботироботи досягнена - функціональні можливості нформаційної технології моделювання ігрової ситуації «Техаський холдем» розширені. Функціональність збільшена в 4,6 рази (14 функцій проти 3).

Дана система в порівняні з проаналізованими покер-системами має переваги в швидкості роботи, простоті візуального інтерфейсу та в параметрах гри, які надають користувачу обирати режим гри, задавати розміри блайндів та розміри стеків обох гравців, цієї можливості немає в жодній із проаналізованих покер-систем.

Розроблена модель системи відрізняється від проаналізованих більшою кількістю підсистем в яких виконуються паралельні розрахунки, за рахунок цього швидкість роботи програми підвищилась в порівняні з проаналізованими.

Розроблена гра досить швидко аналізує ігрову ситуацію та приймає рішення для подальшого ходу. В грі реалізовані два режими: професійний та звичайний. В параметрах гри було додано можливість при звичайному режимі гри задавати стеки обох гравців та блайнди, при чому блайнди та стеки можуть бути різними для обох гравців. Також в звичайному режимі гри додано стрічку, яка містить повну інформацію про ігрову ситуацію гравця, ця стрічка допомагає гравцеві приймати більш оптимальне рішення на кожному етапі гри. В професійному режимі гри блайнди задаються строго по правилам гри, та стеки обох гравців рівні. Довідка в програмі реалізована на трьох мовах, що дасть змогу більшій кількості людей її читати.

Недоліком розробленої гри є те, що гра не може враховувати психологію опонента, тобто не може запам’ятовувати ходи, які були раніше та на основі їх приймати більш оптимальне рішення. Для подолання цього недоліку можна ввести окрему таблицю, в якій будуть міститися ім’я гравця або його нік (для ідентифікації гравця), комбінація з якою грав гравець та опонент, стек обох гравців, та дії, які виконував гравець під час гри. Також недоліком даної гри є, те що гравець може грати тільки проти комп’ютера і не має можливості грати з реальними людьми по інтернету чи локальній мережі. На кожній стадії гри приймаються оптимальні рішення згідно тих розрахунків, які проводяться в головному класі гри «Artificial_intelligence».

4.3 Висновок до розділу 4

У розділі було проведено тестування розробленої програми гри «Техаський холдем» як у звичайному режимі гри, коли користувачу

відображається стрічка, що містить головні параметри ситуації, яку має гравець (в цій стрічці: «Шанси банку», «Потенційні шанси банку», «Шанси на покращення», «Аути», «Математичне сподівання» та «Комбінацію»), так і в професійному режимі гри. В професійному режимі гри даної стрічки немає. Також було протестовано виконання чотирьох типів запитів у турнірній таблиці результатів гри. Було доведено повну працездатність програми та її відповідність поставленому завданню. В результаті аналізу результатів роботи розробленої програми було з`ясовано, що програма-аналог має 3 функції роботи, а розроблена програма має 14 функцій роботи. Тобто мета роботи досягнута – функціональні можливості нформаційної технології моделювання ігрової ситуації «Техаський холдем» розширені. Функціональність збільшена в 4,6 рази (14 функцій проти 3).

5 ЕКОНОМІЧНА ЧАСТИНА

Науково-технічна розробка може бути прийнята та успішно впроваджена, якщо вона відповідає вимогам сучасності в напрямку науково-технічного прогресу та враховує економічні аспекти. Тому необхідно оцінювати економічну ефективність отриманих результатів науково-дослідної роботи.

Магістерська кваліфікаційна робота з розробки та дослідження за темою «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах» відноситься до науково-технічних робіт, які орієнтовані на виведення на ринок (або рішення про виведення науково-технічної розробки на ринок може бути прийнято у процесі проведення самої роботи), тобто коли відбувається так звана комерціалізація науково-технічної розробки. Цей напрямок визначається як пріоритетний, оскільки розроблені результати можуть бути використані різними зацікавленими сторонами, приносячи економічні вигоди. Однак для здійснення цього процесу необхідно знайти потенційного інвестора, який був би зацікавлений у втіленні цього проекту, і переконати його у виправданості вкладання інвестицій в даний проект.

Для цього визначені наступні етапи виконання робіт:

1. Проведено комерційний аудит науково-технічної розробки, що включає в себе визначення науково-технічного рівня та комерційного потенціалу.

2. Розраховані витрати на реалізацію науково-технічної розробки.

3. Проведено розрахунок економічної ефективності науково-технічної розробки в разі її впровадження та комерціалізації потенційним інвестором, а також обґрунтовано економічну доцільність комерціалізації для інвестора.

5.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах» є підвищення реалістичності комп'ютерної ігрової програми, яка реалізує гру справжнього опонента за рахунок використання методів штучного інтелекту.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями, наведеними в табл. 5.1 [19].

Таблиця 5.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в	Технічні та споживчі властивості продукту значно кращі, ніж в
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів

Таблиця 5.1 (Продовження)

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуютьс я у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці.

Таблиця 5.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	Іванчук Я. В., д.т.н. проф. каф. КН, ВНТУ	Колесницький О. К., к.т.н. проф. каф. КН, ВНТУ	Арсенюк І. Р., к.т.н. доц. каф. КН, ВНТУ
	Бали:		
1. Технічна здійсненність концепції	3	3	3
2. Ринкові переваги (наявність аналогів)	4	4	4
3. Ринкові переваги (ціна продукту)	4	3	4
4. Ринкові переваги (технічні властивості)	3	3	2
5. Ринкові переваги (експлуатаційні витрати)	3	3	4
6. Ринкові перспективи (розмір ринку)	2	2	1
7. Ринкові перспективи (конкуренція)	3	3	3
8. Практична здійсненність (наявність фахівців)	2	2	3
9. Практична здійсненність (наявність фінансів)	1	1	1
10. Практична здійсненність (необхідність нових матеріалів)	4	4	4
11. Практична здійсненність (термін реалізації)	3	3	4
12. Практична здійсненність (розробка документів)	4	4	4
Сума балів	$СБ_1$=36	$СБ_2$=35	$СБ_3$=37
Середньоарифметична сума балів $СБ_с$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = \frac{36+35+37}{3} = 36$		

За результатами розрахунків, наведених в таблиці 5.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 5.3 [20].

Таблиця 5.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ ,розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах» становить 36 балів, що, відповідно до таблиці 5.3, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

Результатом роботи є комп’ютерна гра, яка реалізує реального опонента для гри в покер.

Хто може користуватись цими результатами? Результатами може користуватися кінцевий користувач або замовник гри. Гра буде продаватися кінцевому користувачу через мережу інтернет.

В якості аналога для розробки було обрано "PokerStars".

Основні недоліки аналога PokerStars:

- платні деякі функції,
- обмеження для деяких країн,
- мало бонусів для регулярних гравців,
- велика кількість реклами.

У розробці "Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах" дані проблеми вирішуються наступним чином:

- відсутність реклами,
- усі функції безкоштовні,

- в програмі буде два режиму гри, звичайний і професійний, професійний буде передбачати прорахування більших ходів, і буде цікавішим для гравців з досвідом.

Таким чином, інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах буде краще за аналог оскільки є дешевшою з безкоштовними функціями, без реклами і різними рівнями гри.

5.2 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов'язані з проведенням науково-дослідної роботи на тему «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

5.2.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [20]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \qquad (5.1)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=21 дні.

$$З_o = 20000 \cdot 5 / 21 = 4545 \; грн.$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту	20000	909,1	5	4545
Інженер-програміст	50000	2272,7	20	45455
Всього				50000

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_p$) за відповідними найменуваннями робіт НДР на тему «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах» розраховуємо за формулою:

$$З_p = \sum_{i=1}^{n} C_i \cdot t_i , \qquad (5.2)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}} , \qquad (5.3)$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо $Мм$=8000 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б) [20];

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

T_p – середнє число робочих днів в місяці, приблизно T_p = 22 дн;

$t_{зм}$ – тривалість зміни, год.

$C_1 = 8000{,}00 \cdot 1{,}7 \cdot 1{,}65 / (22 \cdot 8) = 127{,}5$ *грн.*

$З_{р1} = 127{,}5 \cdot 25 = 3187{,}5$ *грн.*

Таблиця 5.5 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
1. Розробка програмного коду	30	5	127,5	3187,5
2. Тестування та оптимізація розробленого рішення	20	3	101,3	2025,0
Всього				5212,5

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (5.4)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 11%.

$З_{дод} = (50000+5212{,}5) \cdot 11 / 100\% = 6073{,}38$ *грн.*

5.2.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (5.5)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

$Зн$ = (50000+5212,5+6073,38) · 22 / 100% = 13482,89 грн.

5.2.3 Сировина та матеріали

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах».

Витрати на матеріали ($М$), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{вj}\ , \quad (5.6)$$

де $Н_j$ – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*);

$В_j$ – маса відходів j-го найменування, кг;

$Ц_{вj}$ – вартість відходів j-го найменування, грн/кг.

Проведені розрахунки зведемо до таблиці.

Таблиця 5.6 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Вартість витраченого матеріалу, грн
Папір для записів FAXE 70 A5-250	200	1	200
Органайзер офісний OFFICE 100	220	1	220
Диск оптичний OPTIMA CD	15,5	2	31
Flesh-пам'ять GOODRAM 64 C10A	420	1	420
Всього			871
З врахуванням коефіцієнта транспортування			958,1

5.2.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах» відсутні.

5.2.5 Спецустаткування для наукових (експериментальних) робіт

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення.

В роботі витрати на спецустаткування не передбачені.

5.2.6 Програмне забезпечення для наукових (експериментальних) робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних)

необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (5.7)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, ($K_i = 1,10…1,12$);

k – кількість найменувань програмних засобів.

Оскільки в роботі використовувалися безкоштовні програмні засоби, тому дані витрати не враховані.

5.2.7 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12} , \qquad (5.8)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (45000 \cdot 1) / (2 \cdot 12) = 1875$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 5.7 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Комп'ютер	45000	2	1	1875,00
Принтер	6000	2	1	250,00
Приміщення лабораторії	250000	20	1	1041,67
Всього				3166,67

5.2.8 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \qquad (5.9)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e$ = 7,5 грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$B_e = 0,25 \cdot 240,0 \cdot 7,5 \cdot 0,5 / 0,8 = 281,25$ *грн.*

5.2.9 Службові відрядження

До статті «Службові відрядження» дослідної роботи на тему «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах» належать витрати на відрядження штатних

працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов'язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з'їзди, конференції, наради, пов'язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{Н_{св}}{100\%}, \quad (5.10)$$

де $Н_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св}$ = 20%.

$В_{св}$ = (50000+5212,5) · 20 / 100% = 11042,5 грн.

5.2.10 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» відсутні.

5.2.11 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (5.11)$$

де $H_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$І_в$ = (50000+5212,5) · 50 / 100% = 27606,25 грн.

5.2.12 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{H_{нзв}}{100\%}\ , \qquad (5.12)$$

де $H_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$В_{нзв}$ = (50000+5212,5) · 100 / 100% = 55212,5 грн.

Витрати на проведення науково-дослідної роботи на тему «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах». розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + B_{спец} + B_{прг} + A_{обл} + B_е + B_{св} + B_{сп} + І_в + B_{нзв}. \qquad (5.13)$$

$В_{заг}$=

50000+5212,5+6073,38+13482,89+958,1+3166,67+281,25+11042,5+27606,25

+55212,5 = 173036,03 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{В_{заг}}{\eta}, \quad (5.14)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,9.

ЗВ = 173036,03 / 0,9 = 192262,26 грн.

5.3 Розрахунок економічної ефективності науково-технічної розробки при її можливій комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження проведені за темою «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах» передбачають комерціалізацію протягом 3-х років реалізації на ринку.

В цьому випадку основу майбутнього економічного ефекту будуть формувати:

ΔN – збільшення кількості споживачів яким надається відповідна інформаційна послуга у періоди часу, що аналізуються;

N – кількість споживачів яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково-технічної розробки, приймемо 1 особа

$Ц_б$ – вартість послуги у році до впровадження інформаційної системи, приймемо 350000,00 грн;

$\pm\Delta Ц_o$ – зміна вартості послуги від впровадження результатів, приймемо зростання на 10000,00 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 3-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою [20]:

$$\Delta П_i = (\pm\Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (5.15)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2024 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0,8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту). Приймемо ρ =40%;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2024 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$\Delta П_1 = (1 \cdot 10000 + 350000 \cdot 20) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 1231659,1$ грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2 = (1 \cdot 10000 + 350000 \cdot (20+30)) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 3084877$грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3 = (1 \cdot 10000 + 350000 \cdot (20+30+50)) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 6159754$ грн.

Приведена вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (5.16)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =18%;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = 1231659{,}1 /(1+0{,}18)^1+3084877/(1+0{,}18)^2+6159754 /(1+0{,}18)^3=$$
$$= 6159754 \text{ грн.}$$

Величина початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \qquad (5.17)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$ =2;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 192262,26 грн.

$PV = k_{інв} \cdot ЗВ = 2 \cdot 192262{,}26 = 384524{,}52$грн.

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV \quad (5.18)$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 6159754грн;

PV – теперішня вартість початкових інвестицій, 384524,52 грн.

$E_{абс} = ПП - PV$ = 6159754 - 384524,52 = 6365386,05 грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Тж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (5.19)$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, грн;

PV – теперішня вартість початкових інвестицій, грн;

$T_ж$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 3 роки.

$$E_в = \sqrt[Тж]{1 + \frac{E_{абс}}{PV}} - 1 = (1+6365386,05/384524,52)^{1/3} - 1 = 2,24.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \quad (5.20)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2024 році в Україні d =0,1;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,25.

$\tau_{мін}$ = 0,1+0,25 = 0,35 < 0,6 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \tag{5.21}$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 2,24 = 0,4 р.

$T_{ок}$ < 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

5.4 Висновок до розділу 5

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах» становить 36,0 балів, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього). Також термін окупності становить 0,4

р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок. Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах»

ВИСНОВКИ

В даній магістерській роботі було проведено аналіз предметної області інформаційних технологій моделювання ігрової ситуації в розподілених масштабованих системах, наведено основні визначення предметної області, проаналізовано і подано головні правила гри в «Техаському холдемі», проведено аналіз існуючих покер-систем в «Техаському холдемі» (pokerstars, cristalpalaceonline, erapokera, lasvilis), наведено їх переваги та недоліки. В постановці задачі наведено вхідні, вихідні дані, а також наведені системні вимоги до розроблюваної гри. На основі аналізу недоліків сучасних покер-систем сформульовано мету роботи – розширення функціональних можливостей.

Було наведено обґрунтування форми подання ігрової ситуації в «Техаському холдемі». В якості подання карт було обрано структуру, яка буде містити два поля - масть та ранг карти. Така форма подання карт значно зменшила аналітичні розрахунки та спростила написання програми. Для сортування колоди карт обрано метод Хоара, який забезпечує високу швидкість сортування карт ($\log_2 n^n$). Розроблено математичну модель гри та розроблено процес прийняття рішень в «Техаському холдемі». Запропоновано стратегію гравця в «Техаському холдемі». Стратегія гравця будується на таблицях (певний аналог бази знань), де зазначено можливі ситуації та можливі дії в них, дані таблиці забезпечують доцільність кожного наступного ходу.

Також було розроблено оціночну функцію, яка в свою чергу побудована на основі математичної моделі гри, розробленої в розділі 2. Також розроблено алгоритм інформаційної технології гри в «Техаському холдемі», розроблений алгоритм передбачає моделювання роботи справжнього опонента гравця. На основі вищенаведеної інформації розроблено програмний продукт на об’єктно-орієнтованій мові C#.

На завершення було проведено тестування розробленої програми гри «Техаський холдем» як у звичайному режимі гри, коли користувачу відображається стрічка, що містить головні параметри ситуації, яку має гравець (в цій стрічці: «Шанси банку», «Потенційні шанси банку», «Шанси на покращення», «Аути», «Математичне сподівання» та «Комбінацію»), так і в професійному режимі гри. В професійному режимі гри даної стрічки немає. Також було протестовано виконання чотирьох типів запитів у турнірній таблиці результатів гри. Було доведено повну працездатність програми та її відповідність поставленому завданню. В результаті аналізу результатів роботи розробленої програми було з`ясовано, що програма-аналог має 3 функції роботи, а розроблена програма має 14 функцій роботи. Тобто мета роботи досягнута – функціональні можливості нформаційної технології моделювання ігрової ситуації «Техаський холдем» розширені. Функціональність збільшена в 4,6 рази (14 функцій проти 3).

Напрямком подальших досліджень має бути покращення моделі системи з урахуванням поведінки різних гравців та збільшення щвидкодії роботи системи при великій кількості користувачів.

У економічному розділі було визначено рівень комерційного потенціалу розробки за темою «Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах» становить 36,0 балів, що вказує на комерційну вагомість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього). Термін окупності становить 0,4 р., що менше 3-х років, що говорить про комерційну привабливість науково-технічної розробки і може спонукати потенційних інвесторів на фінансування впровадження цієї розробки та виведення її на ринок.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Загнітко В. М., Колесницький О. К. Математична модель ігрової ситуації в розподілених масштабованих системах. Матеріали LIII науково-технічної конференції підрозділів ВНТУ, Вінниця, 20-22 березня 2024 р. Електрон. текст. дані. 2024. URI: https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2024/paper/view/20853.

2. Загнітко В. М., Колесницький О. К. Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах. Матеріали LIII науково-технічної конференції підрозділів ВНТУ, Вінниця, 20-22 березня 2024 р. Електрон. текст. дані. 2024. URI: https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2024/paper/view/20543.

3. Загнітко В. М., Колесницький О. К. Розробка експертної системи для діагностування інфекційних вірусних захворювань дихальної системи організму людини. Матеріали LIII науково-технічної конференції підрозділів ВНТУ, Вінниця, 20-22 березня 2024 р. Електрон. текст. дані. 2024. URI: https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2024/paper/view/20545.

4. Покер [Електронний ресурс]. – Електрон. текстові дані (402560 байт) – Коваленко О.М., 2012. – Режим доступу: http://www.pokerua.net.

5. Цінність стартових рук у техаському покері [Електронний ресурс]. Електрон. текстові дані (107240 байт) – Іваненко П.В., 2011. – Режим доступу: http://pokerschool.ua/start-hands-value.

6. Куссуль Н. М. Інтелектуальні обчислення: навчальний посібник (із грифом МОН України) / Н. М. Куссуль, А. Ю. Шелестов, А. Н. Лавренюк. - К.: «Наукова думка», 2006. — 186 с. ISBN 966-00-0592-X.

7. Герберт Шилдт. C# 4.0: повне керівництво. / Шилдт Герберт., пер. з англ. К.: ТОВ "ТехноКнига", 2012. - 1056 с.: іл. – Парал. тит. англ.

8. О.М. Колодчак Інтелектуальний аналіз даних Lviv Polytechnic National University Institutional Repository [Електронний ресурс] – Режим доступу: http://ena.lp.edu.ua/bitstream/ntb/26402/1/10_49-58.pdf

9. Математичне сподівання та ймовірність у покері [Електронний ресурс]. Електрон. текстові дані (800123 байт) – Сидоренко Д.В., 2010. – Режим доступу: http://poker-math.com.ua.

10. Месюра В.І., Ваховська Л.М. «Основи проектування систем штучного інтелекту» / В.І. Месюра, Л.М. Ваховська. - В:, ВДТУ, 2000, – 96 с.

11. Smed and Hakonen (2006). Algorithms and Networking for Computer Games [Архівовано 24 листопада 2010 у Wayback Machine.]. John Wiley & Sons. ISBN 0-470-01812-7.

12. Кононова К. Ю. Машинне навчання: методи та моделі: підручник для бакалаврів, магістрів та докторів філософії спеціальності 051 «Економіка» / К. Ю. Кононова. – Харків: ХНУ імені В. Н. Каразіна, 2020. – 301 с.

13. Schaeffer, J.; Burch, N.; Y. Björnsson; Kishimoto, A.; Müller, M.; Lake, R.; Lu, P.; Sutphen, S. (2007). Checkers is Solved (PDF). Science. 317 (5844): 1518—22. doi:10.1126/science.1144079. PMID 17641166.

14. Howard Raiffa Decision Analysis: Introductory Readings on Choices Under Uncertainty. McGraw Hill. 1997. ISBN 0-07-052579-X.

15. Dixit, J. B. (2011-07). Programming in C (англ.). Laxmi Publications. ISBN 978-93-80298-39-9.

16. Т. Кормен; Ч. Лейзерсон; Р. Рівест; К. Стайн (2009). Швидке сортування. Вступ до алгоритмів (вид. 3rd). MIT Press і McGraw-Hill. ISBN 0-262-03384-4.

17. Пасічник В. В. , Шаховська Н.Б. Сховища даних: Навчальний посібник. – Львів: «Магнолія 2006», 2020. – 492 с..

18. Іванчук Я. В., Месюра В. І., Яровий А. А., Манжілевський О. Д. Інтелектуальний аналіз даних та машинне навчання. Частина 1. Базові методи та засоби аналізу даних : навчальний посібник, Вінниця : ВНТУ, 2021. – 69 с., https://iq.vntu.edu.ua/card.php?id=6030

19. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

20. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа – Вінниця : ВНТУ, 2016. – 113 с.

21. Методичні вказівки до виконання магістерських кваліфікаційних робіт для студентів спеціальності 122 «Комп’ютерні науки» [Електронний ресурс] / уклад.: А. А. Яровий, О. К. Колесницький. – Вінниця : ВНТУ, 2023. – (58 с.)

Додаток А (обов'язковий)

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія моделювання ігрової ситуації в розподілених масштабованих системах

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ кафедра комп'ютерних наук, ФІТА
(кафедра, факультет)

Показники звіту подібності Turnitin

Оригінальність 86,00% Схожість 14,00%

Аналіз звіту подібності (відмітити потрібне):

✓ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Ознайомлені з повним звітом подібності, який був згенерований системою Turnitin щодо роботи.

Автор роботи Загнітко В.М.

Керівник роботи Колесницький О.К.

Опис прийнятого рішення

Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку Озеранський В.С.

Додаток Б (обов'язковий)

Лістинг програми

```
 class Greedy_algorithm : Game_Poker
{
   public ArrayList Greedy(int chyslo_fishok)//Цей алгоритм підраховує, які фішки потрібно взяти для того, щоб скласти необхідну суму
   {
      int n = chyslo_fishok;
      int[] mass = new int[] { 1000, 500, 100, 25, 5, 1 };
      ArrayList mass_fishka = new ArrayList();
      int max = mass[0], i = 0;
      for (; n != 0; )
      {
         for (i = 0; i < mass.Length - 1; )//5
         {
            if (n >= max)
            {
               n -= max;
               mass_fishka.Add(max);
            }
            else
               max = mass[i + 1];  i++;
      }
      return mass_fishka;
   }
}

class Game : Game_Poker
{
   //Колода карт
   char[] Mast_kart = new char[4] { 'S', 'H', 'D', 'C' };
   char[] Rang_kart = new char[4] { 'J', 'Q', 'K', 'A' };
   char Ten = 'T'; //Представлення десятки
   int count_cards = 0;
   Random rand = new Random(); //Псевдовипадкові числа
   //Ініціалізація карт
   public Game()
   {
      for (int i = 0; i < 4; i++)
      {
         for (int j = 2; j < 15; j++)
         {
            if (j < 11)
            {
               if (j == 10)
                  card[count_cards].Rang = Ten;
               else
                  card[count_cards].Rang = Convert.ToChar(j + '0');
            }
            else
               card[count_cards].Rang = Rang_kart[j - 11];
            card[count_cards].Mast = Mast_kart[i];
            count_cards++;
         }
      }
   }
   //Сортування карт (індексів)
   public int[] Sort_karty(out string str)// Потім потрібно буде видалити аут
   {
      int[] card_mass = new int[52];///Масив рангів карт
      str = "";
      for (int i = 0; i < card_mass.Length; i++)
```

```
            card_mass[i] = i;
        sorting(card_mass, 0, 51);///Сортування індексів масиву карт
        for (int i = 0; i < 9; i++)
            str += (card[card_mass[i]].Mast + "" + card[card_mass[i]].Rang + " ");
        return card_mass;
    }
    //Можна буде потім видалити
    public string Write_cards()
    {
        string str = "";
        for (int i = 0; i < card.Length; i++)
            str += (card[i].Mast + "" + card[i].Rang + " ");
        return str;
    }
    //Метод для сортування карт на базі методу Хоара
    private void sorting(int[] arr, int first, int last)
    {
        Random rand = new Random();
        //int p = arr[(last - first) / 2 + first] + rand.Next(26) + 1;
        int temp;
        int i = first, j = last;
        while (i <= j)
        {
            int p = arr[(last - rand.Next(i ^ rand.Next(1))) / 2 + rand.Next(2 ^ rand.Next(1))] + rand.Next(i ^ rand.Next(1))
+ rand.Next(2 ^ rand.Next(1));
            switch ((j + rand.Next(i)) % (rand.Next(j) + rand.Next(i) + rand.Next(5) + 1) ^ rand.Next(1))
            {
                case 1:
                    {
                        while (arr[i] < rand.Next(j ^ rand.Next(1)) && i < last) ++i;
                        while (arr[j] > rand.Next(i ^ rand.Next(1)) && j > first) --j;
                        break;
                    }
                case 2:
                    {
                        while (arr[i] < p && i < last) ++i;
                        while (arr[j] < p && j > first) --j;
                        break;
                    }
                case 3:
                    {
                        while (arr[i] > rand.Next(j ^ rand.Next(1)) && i < last) ++i;
                        while (arr[j] < rand.Next(i ^ rand.Next(1)) && j > first) --j;
                        break;
                    }
                case 4:
                    {
                        while (arr[i] > p && i < last) ++i;
                        while (arr[j] > p && j > first) --j;
                        break;
                    }
            }
            for (int k = 0; k < arr.Length / 2; k++)
            {
                temp = arr[k];
                arr[k] = arr[arr.Length - k - 1];
                arr[arr.Length - k - 1] = temp;
            }
            if (i <= j)
            {
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
                ++i; --j;
            }
        }
```

```
            if (j > first) sorting(arr, first, j);
            if (i < last) sorting(arr, i, last);
        }
    }

    class Create_Hash_Table_Outs : Game_Poker
    {
        public Hashtable Create_Hash_Table(Hashtable Hash_table_Outs, string T_Outs)
        {
            string[] mass_Outs_Oponent = new string[5];
            string str_keys = "";
            string str_values = "";
            int count_mass_Outs_Oponent = 0;//Лічильник для масиву mass_Outs_Oponent
            for (int i = 0; i < T_Outs.Length; i++)
            {
                if (T_Outs[i] == ';')
                    count_mass_Outs_Oponent++;
                if (T_Outs[i] == ';' || T_Outs[i] == ' ')
                    continue;
                else
                    mass_Outs_Oponent[count_mass_Outs_Oponent] += T_Outs[i];
            }
            for (int i = 0; i < mass_Outs_Oponent.Length; i++)
            {
                str_keys = "";
                str_values = "";
                for (int j = 0; j < mass_Outs_Oponent[i].Length; j++)
                {
                    if ((mass_Outs_Oponent[i][j] >= '0' && mass_Outs_Oponent[i][j] <= '9') || mass_Outs_Oponent[i][j] == ',')
                        str_values += mass_Outs_Oponent[i][j];
                    if (mass_Outs_Oponent[i][j] != ':' && !(mass_Outs_Oponent[i][j] >= '0' && mass_Outs_Oponent[i][j] <=
'9')
                        && mass_Outs_Oponent[i][j] != ',' && mass_Outs_Oponent[i][j] != '=')
                    {
                        if (mass_Outs_Oponent[i][j] == '_')
                            str_keys += " ";
                        else
                            str_keys += mass_Outs_Oponent[i][j];
                    }
                }
                Hash_table_Outs.Add(str_keys, str_values);
            }
            ICollection I_Hash_table_Outs_Oponent = Hash_table_Outs.Keys;
            return Hash_table_Outs;
        }
    }

    class Artificial_intelligence : Game_Poker
    {
        string odds_stage = "";//Шанси банка на кожному кроці гри
        int kilkist_kart_v_kolodi = 0;//Кількість карт в колоді
        int i_strength_of_hands = 0;//Сила руки
        int rate_game = 0;//Ставка гравця (РЕЙЗ)

        public Artificial_intelligence(int table)
        {
            switch (table)
            {
                case 1:
                    {
                        odds_stage = "Flop to Tern";
                        kilkist_kart_v_kolodi = 47;
                        break;
                    }
                case 2:
                    {
```

```
                odds_stage = "Tern to River";
                kilkist_kart_v_kolodi = 46;
                break;
            }
        default:
            {
                odds_stage = "Flop River";
                kilkist_kart_v_kolodi = 45;
                break;
            }
    }
}

public Artificial_intelligence(int rate_game, int table)
{
    switch (table)
    {
        case 1:
            {
                odds_stage = "Flop to Tern";
                kilkist_kart_v_kolodi = 47;
                break;
            }
        case 2:
            {
                odds_stage = "Tern to River";
                kilkist_kart_v_kolodi = 46;
                break;
            }
        default:
            {
                odds_stage = "Flop River";
                kilkist_kart_v_kolodi = 45;
                break;
            }
    }
    this.rate_game = rate_game;
}

//Даний метод обчислює актуальність даної ставки після префлопа
public string Acquittal_rate(Hashtable Hash_Table_Outs, Hashtable Hash_Table_Drop, int rate, out int
i_strength_of_hand, int Bank, out double[] artificial_intelligence, bool b_player, out int i_whole_rate)
{
    ICollection I_Hash_Table_Outs = Hash_Table_Outs.Keys;
    ArrayList arr_mass_rate = new ArrayList();//Цей масив буде містити значення ставок, що зробив
ОПОНЕНТ
    ArrayList arr_mass_strong_hand = new ArrayList();//Цей масив буде містити значення сили руки
    //int? i_selected_item = 0;//Необхідний для Combobox
    double odds = 0.0;
    double pot_odds = 0.0;
    double expected_value = 0.0;
    double outs = 0;
    double call_bank = 0.0;//Дана змінна буде містити відношення банку на 100 (bank/100), і це число буде
множитися на Call, щоб ставка була залежна від велечини банку
    double acquittal_rate = 0.0;//Актуальність ставки
    double chances_for_improvment = 0.0;//Дана змінна буде містити шанси на покращення існуючої
комбінації
    double gain_is = 0.0;//Дана змінна буде містити приблизний вийграш
    double the_loss_will_be = 0.0;//Дана змінна буде містити приблизний пройграш
    double temp_acquittal_rate_call_bank = 0.0;
    int i_selected_item_combobox = 0;
    koeficient = stack.Stack_gamer / 100;
    i_Big_Blind = stack.Stack_gamer / 10;
    i_Small_Blind = stack.Stack_gamer / 20;
    i_strength_of_hand = 0;//Дана змінна буде містити силу руки в числовому евіваленті
    artificial_intelligence = new double[5];
```

```
            i_whole_rate = 0;
            bool b_call = false;//Ця булівська змінна буде містити доцільність колірования
            string str_combination_hashtable = "";
            ICollection c = Hash_Table_Drop.Keys;
            foreach (string st in c)
               str_combination_hashtable = (string)Hash_Table_Drop[st];
            foreach (string stage in I_Hash_Table_Outs)
            {
               if (odds_stage == stage)//Якщо знайшли в хеш таблиці значення, яке відповідає стадії гри
                  odds = Convert.ToDouble(Hash_Table_Outs[stage]);//Присвоюємо шанси банка змінній
               if ("Outs" == stage)//Знаходимо кількість аутів в нашій Хеш таблиці
                  outs = Convert.ToDouble(Hash_Table_Outs[stage]);//Присвоємо змінній outs кількість аутів для
найкращої комбінації
            }
            chances_for_improvment = Chances_for_improvement(outs);
            i_strength_of_hand = Strength_of_hand(Hash_Table_Drop);//Цей метод визначає силу руки
            i_strength_of_hands = i_strength_of_hand;
            if (str_combination_hashtable.Equals("Straight or Flush"))
               i_strength_of_hand = 7000;
            if (str_combination_hashtable.Equals("Flash dro"))
               i_strength_of_hand = 6200;
            if (str_combination_hashtable.Equals("Straight dro"))
               i_strength_of_hand = 5800;
            b_call = Call(out gain_is, out the_loss_will_be, outs, rate);//Булівська змінна буде містити значення чи є сенс
колувати ставку
            expected_value = Expected_value(outs);//Підраховує математичне сподівання
            if (!b_call)
               expected_value *= (-1);
            if (str_combination_hashtable.Equals("Straight or Flush") || str_combination_hashtable.Equals("Flash dro") ||
str_combination_hashtable.Equals("Straight dro"))
               expected_value *= (-1);
            acquittal_rate = (chances_for_improvment * i_strength_of_hand * expected_value);//Актуальність ставки
            call_bank = Bank / koeficient * 10;
            pot_odds = Convert.ToDouble(rate) / Convert.ToDouble(Bank);
            artificial_intelligence[0] = odds;//Записуємо шанси банку в масив
            artificial_intelligence[1] = pot_odds;//Записуємо потенційні шанси банку в масив
            artificial_intelligence[2] = chances_for_improvment;//Записуємо шанси на покращення
            artificial_intelligence[3] = outs;//Записуємо аути в масив
            artificial_intelligence[4] = expected_value;//Записуємо математичне сподівання в масив

            if (b_player)
               i_selected_item_combobox = rate;
            if ((chances_for_improvment + expected_value > pot_odds) && b_call && b_player)//Якщо шанси на
покращення вищі ніж шанси банка, то ставка виправдана, інакше ні
            {
               double temp = acquittal_rate * call_bank * expected_value;
               if (str_combination_hashtable.Equals("Straight or Flush") || str_combination_hashtable.Equals("Flash dro") ||
str_combination_hashtable.Equals("Straight dro"))
                  i_strength_of_hand = 0;
               if (str_combination_hashtable.Equals("Kiker") && i_selected_item_combobox == 0)
                  return "Чек";
               else
               {
                  if (str_combination_hashtable.Equals("Kiker") && i_selected_item_combobox != 0)
                     return "Фолд";
               }
               //Якщо поставив опонент поставив ставку і гравець її підвищив, то рахуємо доцільність колу
               if (rate != 0)
               {
                  b_call = Call(out gain_is, out the_loss_will_be, outs, rate);//Булівська змінна буде містити значення чи є
сенс колувати ставку
                  expected_value = Expected_value(outs);//Підраховує математичне сподівання
                  if (!b_call)
                     expected_value *= (-1);
                  if ((chances_for_improvment + expected_value > pot_odds) && b_call)
                     return "Колл:=" + rate;//The_whole_rate(Convert.ToInt32(acquittal_rate * call_bank)); acquittal_rate
```

```
                else
                    return "Фолд";
                }

                if (i_selected_item_combobox == 0 && acquittal_rate >= i_Big_Blind)//Якщо актуальність ставки більша ніж високий блайнд, то заходимо в вітку умови (Якщо доцільно робити колл)
                {
                    if (Convert.ToInt32(acquittal_rate * call_bank * expected_value) > stack.Stack_oponent / 5 && acquittal_rate >= i_Big_Blind)//Convert.ToInt32(acquittal_rate * call_bank) > stack.Stack_oponent / 5 &&
                    {
                        while (temp >= i_Big_Blind && temp >= stack.Stack_oponent)
                            temp /= 10;
                    }
                    temp /= koeficient;
                    temp += i_strength_of_hand / koeficient;
                    if (i_mass_count > 0 && ((int)arr_mass_rate[i_mass_count] <= temp || (int)arr_mass_strong_hand[i_mass_count] <= i_strength_of_hand))
                    {
                        if (temp < i_Big_Blind)//Якщо ставка меньше ніж великий блайнд, то робимо її як великий блайнд
                            temp += (i_Big_Blind - temp);
                        if (temp > stack.Stack_gamer)
                            temp = i_Big_Blind;
                        i_whole_rate = The_whole_rate(Convert.ToInt32(temp));//acquittal_rate
                        arr_mass_rate.Add(i_whole_rate);//Записуємо ставки опонента
                        arr_mass_strong_hand.Add(i_strength_of_hand);
                        i_mass_count++;
                        return "Колл:=" + The_whole_rate(Convert.ToInt32(temp));//The_whole_rate(Convert.ToInt32(acquittal_rate * call_bank)); acquittal_rate
                    }
                    else
                    {
                        if (i_mass_count == 0 && rate==0)
                        {
                            if (temp < i_Big_Blind)//Якщо ставка меньше ніж великий блайнд, то робимо її як великий блайнд
                                temp += (i_Big_Blind - temp);
                            if (temp > stack.Stack_gamer)
                                temp = i_Big_Blind;
                            i_whole_rate = The_whole_rate(Convert.ToInt32(temp));//acquittal_rate
                            arr_mass_rate.Add(i_whole_rate);//Записуємо ставки опонента
                            arr_mass_strong_hand.Add(i_strength_of_hand);
                            i_mass_count++;
                            return "Колл:=" + The_whole_rate(Convert.ToInt32(temp));//The_whole_rate(Convert.ToInt32(acquittal_rate * call_bank)); acquittal_rate
                        }
                        else
                        {
                            if (rate == 0 && (!b_call || expected_value <= 0))
                            {
                                arr_mass_rate.Add(i_whole_rate);//Записуємо ставки опонента
                                arr_mass_strong_hand.Add(i_strength_of_hand);
                                i_mass_count++;
                                return "Чек";
                            }
                            else
                                return "Фолд";
                        }
                    }

                }
                else
                {
                    if (i_selected_item_combobox > 0 && acquittal_rate * call_bank > i_selected_item_combobox)
                    {
                        if (Convert.ToInt32(acquittal_rate * call_bank) > stack.Stack_oponent)
                        {
```

```
                    while (Convert.ToInt32(acquittal_rate * call_bank * expected_value) > stack.Stack_oponent)
                    {
                        temp_acquittal_rate_call_bank = acquittal_rate;
                        temp_acquittal_rate_call_bank /= 10;
                        acquittal_rate = temp_acquittal_rate_call_bank;
                    }
                }
                i_whole_rate = The_whole_rate(Convert.ToInt32(acquittal_rate * call_bank));
                return "Рейз:=" + The_whole_rate(Convert.ToInt32(acquittal_rate * call_bank));
            }
        }
    }
    else
        if (i_selected_item_combobox == 0 && b_player)//Якщо гравець поставив ставку, то Чек або Фолд
            return "Чек";
        else
        {
            if (b_player)
                return "Фолд";
        }
    i_whole_rate = 0;
    return "";//Якщо грає звичайний гравець, то просто повертаємо пусту стрічку
}

//Даний метод обчислює актуальність ставки на префлопі
public string String_call_preflop(string str_Ruka, int rate)
{
    int[] mass_Ruka = new int[2];
    int i_Ruka = 0;
    int i_temp = 0;
    bool mast = false;//Якщо карти одномастні, то ця змінна буде містити істину, інакше хибність
    string str_sort_cards_preflop_oponent = "";//Ця стрічка буде містити дві карти, які відсортовані від більшої
до меньшої, щоб виконати запит
    string[] str_mass_Ruka = new string[2];
    for (int i = 1; i < 5; i += 3)
    {
        for (int j = 0; j < c_Rang_mass.Length; j++)
        {
            if (str_Ruka[i] == c_Rang_mass[j])
            {
                str_mass_Ruka[i_Ruka] += str_Ruka[i - 1];
                str_mass_Ruka[i_Ruka] += str_Ruka[i];
                mass_Ruka[i_Ruka] = j;
                i_Ruka++;
            }
        }
    }
    if (mass_Ruka[0] < mass_Ruka[1])
    {
        i_temp = mass_Ruka[0];
        mass_Ruka[0] = mass_Ruka[1];
        mass_Ruka[1] = i_temp;
    }
    for (int i = 0; i < mass_Ruka.Length; i++)
    {
        for (int j = 0; j < str_mass_Ruka.Length; j++)
        {
            if (str_mass_Ruka[j][1] == c_Rang_mass[mass_Ruka[i]])
                str_sort_cards_preflop_oponent += str_mass_Ruka[j][1];//Дана стрічка буде містити ранги карт
ОПОНЕНТА
            }
    }
    if (str_mass_Ruka[0][0] == str_mass_Ruka[1][0])
        mast = true;
    if (mast)
        str_sort_cards_preflop_oponent += " ";
```

```
            else
                str_sort_cards_preflop_oponent += "s";
            return str_sort_cards_preflop_oponent;
        }

        //Даний метод підраховує шанси на покращення комбінації
        private double Chances_for_improvement(double outs)
        {
            return Math.Round((outs / (kilkist_kart_v_kolodi - outs)), 2);
        }

        //Даний метод підраховує доцільність колірования, при цьоу обраховує можливий виграш і програш
        private bool Call(out double gain_is, out double the_loss_will_be, double outs, int rate)
        {
            gain_is = 0.0;
            the_loss_will_be = 0.0;
            gain_is = outs * bank.The_Bank + i_strength_of_hands;//Можливий виграш
            the_loss_will_be = (kilkist_kart_v_kolodi - outs) * rate;
            if (gain_is - the_loss_will_be > 0)
                return true;
            else
                return false;
        }

        //Даний метод підраховує математичне очікування
        private double Expected_value(double outs)
        {
            double Expected_value = 0.0;//Математичне очікування
            switch (odds_stage)
            {
                case "Flop River":
                    {
                        Expected_value = 1 - ((1 - (outs / 47)) * (1 - (outs / 46)));
                        break;
                    }
                default:
                    {
                        Expected_value = outs / kilkist_kart_v_kolodi;
                        break;
                    }
            }
            return Math.Round((Expected_value * 100), 2);
        }

        //Даний метод визначає силу руки
        private int Strength_of_hand(Hashtable Hash_Table_Drop)
        {
            string[] str_combination_poker = new string[10] { "Royal Flush", "Straight flush", "4 of kind", "Full house",
"Flush", "Straight", "3 of kind", "2 Pair", "1 Pair", "Kiker" };//DELETE
            ICollection I_Hash_Table_Drop = Hash_Table_Drop.Keys;
            string str_combination_name = "";//Дана змінна буде містити назву комбінації
            string str_cards = "";//Дана змінна буде містити найкращі 5 карт
            string str_temp = "";//Дана змінна буде тимчасово зберігати карти
            int i_strength_of_hand = 0;//Дана змінна буде містити силу руки в числовому евіваленті
            int i_index_v_kolodi_kart = 0;//Дана змінна буде містити індекс найкращої карти в колоді карт
            foreach (string combination in I_Hash_Table_Drop)
            {
                str_combination_name = Hash_Table_Drop[combination].ToString();//Ця змінна буде містити комбінацію,
яка є
                str_cards = combination;//Дана змінна буде містити найкращі 5 карт
            }
            switch (str_combination_name)
            {
                case "Kiker":
                    {
                        i_strength_of_hand = Strength_of_hand_kiker(Hash_Table_Drop) * 10;
```

```
                    break;
                }
            case "1 Pair":
                {
                    int[] kilkist_vhodjen_v_best_combination = new int[5];//Даний масив буде містити кількість входжень елементів в стрічці з найкращою комбінацією
                    for (int i = 1, k = 0; k < kilkist_vhodjen_v_best_combination.Length; i += 3, k++)
                        kilkist_vhodjen_v_best_combination[k] = HowOccurrences(str_cards, str_cards[i]);
                    for (int i = 0; i < kilkist_vhodjen_v_best_combination.Length; i++)
                    {
                        if (kilkist_vhodjen_v_best_combination[i] == 2)
                        {
                            i_index_v_kolodi_kart = i;
                            break;
                        }
                    }
                    i_index_v_kolodi_kart *= 3;//Так як крок складає 3, то множимо на 3
                    str_temp += str_cards[i_index_v_kolodi_kart];//Додаємо в тимчасову змінну карту, яка має змінну, для того порахувати силу пари
                    str_temp += str_cards[i_index_v_kolodi_kart + 1];
                    for (int i = 0; i < 1; i++)
                    {
                        for (int j = 0; j < c_Rang_mass.Length; j++)
                        {
                            if (str_temp[1] == c_Rang_mass[j])
                            {
                                i_index_v_kolodi_kart = j;
                                break;
                            }
                        }
                    }
                    i_index_v_kolodi_kart++;//Інкрементуємо, тому що масив в C# починається з нуля
                    i_strength_of_hand = 1300 + 2 * i_index_v_kolodi_kart + Strength_of_hand_kiker(Hash_Table_Drop);
                    break;
                }
            case "2 Pair":
                {
                    int[] kilkist_vhodjen_v_best_combination = new int[5];//Даний масив буде містити кількість входжень елементів в стрічці з найкращою комбінацією
                    int i_index_v_kolodi_kart_1 = -1;//Для першої пари
                    int i_index_v_kolodi_kart_2 = -1;//Для другої пари
                    for (int i = 1, k = 0; k < kilkist_vhodjen_v_best_combination.Length; i += 3, k++)
                        kilkist_vhodjen_v_best_combination[k] = HowOccurrences(str_cards, str_cards[i]);
                    for (int i = 0; i < kilkist_vhodjen_v_best_combination.Length; i++)
                    {
                        if (kilkist_vhodjen_v_best_combination[i] == 2)
                        {
                            if (i_index_v_kolodi_kart_1 == -1)
                                i_index_v_kolodi_kart_1 = i;
                            else
                                i_index_v_kolodi_kart_2 = i;
                            kilkist_vhodjen_v_best_combination[i] = 0;
                        }
                    }
                    i_index_v_kolodi_kart_1 *= 3;//Так як крок складає 3, то множимо на 3
                    i_index_v_kolodi_kart_2 *= 3;//Так як крок складає 3, то множимо на 3
                    for (int i = 0; i < 2; i++)
                    {
                        if (i == 0)
                        {
                            str_temp += str_cards[i_index_v_kolodi_kart_1];//Додаємо в тимчасову змінну карту, яка має змінну, для того порахувати силу пари
                            str_temp += str_cards[i_index_v_kolodi_kart_1 + 1];
                            str_temp += " ";
                            for (int j = 0; j < c_Rang_mass.Length; j++)
                            {
```

```
                        if (str_temp[1] == c_Rang_mass[j])
                        {
                            i_index_v_kolodi_kart_1 = j;
                            break;
                        }
                    }
                }
                else
                {
                    str_temp += str_cards[i_index_v_kolodi_kart_2 - 1];//Додаємо в тимчасову змінну карту, яка має змінну, для того порахувати силу пари
                    str_temp += str_cards[i_index_v_kolodi_kart_2];
                    for (int j = 0; j < c_Rang_mass.Length; j++)
                    {
                        if (str_temp[4] == c_Rang_mass[j])
                        {
                            i_index_v_kolodi_kart_2 = j;
                            break;
                        }
                    }
                }
                i_index_v_kolodi_kart_1++;
                i_index_v_kolodi_kart_2++;
                i_strength_of_hand = 2600 + 2 * i_index_v_kolodi_kart_1 + 2 * i_index_v_kolodi_kart_2 + Strength_of_hand_kiker(Hash_Table_Drop);
            }
            break;
        }
    case "3 of kind":
        {
            int[] kilkist_vhodjen_v_best_combination = new int[5];//Даний масив буде містити кількість входжень елементів в стрічці з найкращою комбінацією
            int i_index_v_kolodi_kart_1 = -1;//Для першої карти сета
            int i_index_v_kolodi_kart_2 = -1;//Для другої карти сета
            int i_index_v_kolodi_kart_3 = -1;//Для третьої карти сета
            for (int i = 1, k = 0; k < kilkist_vhodjen_v_best_combination.Length; i += 3, k++)
                kilkist_vhodjen_v_best_combination[k] = HowOccurrences(str_cards, str_cards[i]);
            for (int i = 0; i < kilkist_vhodjen_v_best_combination.Length; i++)
            {
                if (kilkist_vhodjen_v_best_combination[i] == 3)
                {
                    if (i_index_v_kolodi_kart_1 == -1)
                        i_index_v_kolodi_kart_1 = i;
                    else
                    {
                        if (i_index_v_kolodi_kart_2 == -1)
                            i_index_v_kolodi_kart_2 = i;
                        else
                            i_index_v_kolodi_kart_3 = i;
                    }
                    kilkist_vhodjen_v_best_combination[i] = 0;
                }
            }
            i_index_v_kolodi_kart_1 *= 3;//Так як крок складає 3, то множимо на 3
            i_index_v_kolodi_kart_2 *= 3;//Так як крок складає 3, то множимо на 3
            i_index_v_kolodi_kart_3 *= 3;//Так як крок складає 3, то множимо на 3
            for (int i = 0; i < 3; i++)
            {
                if (i == 0)
                {
                    str_temp += str_cards[i_index_v_kolodi_kart_1];//Додаємо в тимчасову змінну карту, яка має змінну, для того порахувати силу пари
                    str_temp += str_cards[i_index_v_kolodi_kart_1 + 1];
                    str_temp += " ";
                    for (int j = 0; j < c_Rang_mass.Length; j++)
                    {
```

```
                        if (str_temp[1] == c_Rang_mass[j])
                        {
                            i_index_v_kolodi_kart_1 = j;
                            break;
                        }
                    }
                }
                else
                {
                    if (i == 1)
                    {
                        str_temp += str_cards[i_index_v_kolodi_kart_2 - 1];//Додаємо в тимчасову змінну карту, яка має змінну, для того порахувати силу пари
                        str_temp += str_cards[i_index_v_kolodi_kart_2];
                        str_temp += " ";
                        for (int j = 0; j < c_Rang_mass.Length; j++)
                        {
                            if (str_temp[4] == c_Rang_mass[j])
                            {
                                i_index_v_kolodi_kart_2 = j;
                                break;
                            }
                        }
                    }
                    else
                    {
                        str_temp += str_cards[i_index_v_kolodi_kart_3];//Додаємо в тимчасову змінну карту, яка має змінну, для того порахувати силу пари
                        str_temp += str_cards[i_index_v_kolodi_kart_3 + 1];
                        for (int j = 0; j < c_Rang_mass.Length; j++)
                        {
                            if (str_temp[7] == c_Rang_mass[j])
                            {
                                i_index_v_kolodi_kart_3 = j;
                                break;
                            }
                        }
                    }
                }
                i_index_v_kolodi_kart_1++;
                i_index_v_kolodi_kart_2++;
                i_index_v_kolodi_kart_3++;
                i_strength_of_hand = 3900 + 2 * i_index_v_kolodi_kart_1 + 2 * i_index_v_kolodi_kart_2 + i_index_v_kolodi_kart_3 + Strength_of_hand_kiker(Hash_Table_Drop);
            }
            break;
        }
    case "Straight":
        {
            str_temp += str_cards[0];
            str_temp += str_cards[1];
            for (int i = 0; i < 1; i++)
            {
                for (int j = 0; j < c_Rang_mass.Length; j++)
                {
                    if (str_temp[1] == c_Rang_mass[j])
                    {
                        i_index_v_kolodi_kart = j;
                        break;
                    }
                }
            }
            i_index_v_kolodi_kart++;//Інкрементуємо, тому що масив в C# починається з нуля
            i_strength_of_hand = 5200 + i_index_v_kolodi_kart + Strength_of_hand_kiker(Hash_Table_Drop);
            break;
        }
```

```
                case "Flush":
                    {
                        str_temp += str_cards[0];
                        str_temp += str_cards[1];
                        for (int i = 0; i < 1; i++)
                        {
                            for (int j = 0; j < c_Rang_mass.Length; j++)
                            {
                                if (str_temp[1] == c_Rang_mass[j])
                                {
                                    i_index_v_kolodi_kart = j;
                                    break;
                                }
                            }
                        }
                        i_index_v_kolodi_kart++;//Інкрементуємо, тому що масив в C# починається з нуля
                        i_strength_of_hand = 6500 + i_index_v_kolodi_kart + Strength_of_hand_kiker(Hash_Table_Drop);
                        break;
                    }
                case "Full house":
                    {
                        int[] kilkist_vhodjen_v_best_combination = new int[5];//Даний масив буде містити кількість входжень елементів в стрічці з найкращою комбінацією
                        int i_index_v_kolodi_kart_1 = -1;//Для першої карти сета
                        int i_index_v_kolodi_kart_2 = -1;//Для другої карти сета
                        int i_index_v_kolodi_kart_3 = -1;//Для третьої карти сета
                        int i_index_v_kolodi_kart_1_para = -1;//Для першої пари
                        int i_index_v_kolodi_kart_2_para = -1;//Для другої пари
                        for (int i = 1, k = 0; k < kilkist_vhodjen_v_best_combination.Length; i += 3, k++)
                            kilkist_vhodjen_v_best_combination[k] = HowOccurrences(str_cards, str_cards[i]);
                        for (int i = 0; i < kilkist_vhodjen_v_best_combination.Length; i++)
                        {
                            if (kilkist_vhodjen_v_best_combination[i] == 3)
                            {
                                if (i_index_v_kolodi_kart_1 == -1)
                                    i_index_v_kolodi_kart_1 = i;
                                else
                                {
                                    if (i_index_v_kolodi_kart_2 == -1)
                                        i_index_v_kolodi_kart_2 = i;
                                    else
                                        i_index_v_kolodi_kart_3 = i;
                                }
                                kilkist_vhodjen_v_best_combination[i] = 0;
                            }
                        }
                        i_index_v_kolodi_kart_1 *= 3;//Так як крок складає 3, то множимо на 3
                        i_index_v_kolodi_kart_2 *= 3;//Так як крок складає 3, то множимо на 3
                        i_index_v_kolodi_kart_3 *= 3;//Так як крок складає 3, то множимо на 3
                        for (int i = 0; i < 3; i++)
                        {
                            if (i == 0)
                            {
                                str_temp += str_cards[i_index_v_kolodi_kart_1];//Додаємо в тимчасову змінну карту, яка має змінну, для того порахувати силу пари
                                str_temp += str_cards[i_index_v_kolodi_kart_1 + 1];
                                str_temp += " ";
                                for (int j = 0; j < c_Rang_mass.Length; j++)
                                {
                                    if (str_temp[1] == c_Rang_mass[j])
                                    {
                                        i_index_v_kolodi_kart_1 = j;
                                        break;
                                    }
                                }
                            }
                        }
```

```
                    else
                    {
                        if (i == 1)
                        {
                            str_temp += str_cards[i_index_v_kolodi_kart_2 - 1];//Додаємо в тимчасову змінну карту, яка має
змінну, для того порахувати силу пари
                            str_temp += str_cards[i_index_v_kolodi_kart_2];
                            str_temp += " ";
                            for (int j = 0; j < c_Rang_mass.Length; j++)
                            {
                                if (str_temp[4] == c_Rang_mass[j])
                                {
                                    i_index_v_kolodi_kart_2 = j;
                                    break;
                                }
                            }
                        }
                        else
                        {
                            str_temp += str_cards[i_index_v_kolodi_kart_3];//Додаємо в тимчасову змінну карту, яка має
змінну, для того порахувати силу пари
                            str_temp += str_cards[i_index_v_kolodi_kart_3 + 1];
                            for (int j = 0; j < c_Rang_mass.Length; j++)
                            {
                                if (str_temp[7] == c_Rang_mass[j])
                                {
                                    i_index_v_kolodi_kart_3 = j;
                                    break;
                                }
                            }
                        }
                    }
                    i_index_v_kolodi_kart_1++;
                    i_index_v_kolodi_kart_2++;
                    i_index_v_kolodi_kart_3++;
                }
                str_temp = "";
                for (int i = 1, k = 0; k < kilkist_vhodjen_v_best_combination.Length; i += 3, k++)
                    kilkist_vhodjen_v_best_combination[k] = HowOccurrences(str_cards, str_cards[i]);
                for (int i = 0; i < kilkist_vhodjen_v_best_combination.Length; i++)
                {
                    if (kilkist_vhodjen_v_best_combination[i] == 2)
                    {
                        if (i_index_v_kolodi_kart_1_para == -1)
                            i_index_v_kolodi_kart_1_para = i;
                        else
                            i_index_v_kolodi_kart_2_para = i;
                        kilkist_vhodjen_v_best_combination[i] = 0;
                    }
                }
                i_index_v_kolodi_kart_1_para *= 3;//Так як крок складає 3, то множимо на 3
                i_index_v_kolodi_kart_2_para *= 3;//Так як крок складає 3, то множимо на 3
                for (int i = 0; i < 2; i++)
                {
                    if (i == 0)
                    {
                        str_temp += str_cards[i_index_v_kolodi_kart_1_para];//Додаємо в тимчасову змінну карту, яка має
змінну, для того порахувати силу пари
                        str_temp += str_cards[i_index_v_kolodi_kart_1_para + 1];
                        str_temp += " ";
                        for (int j = 0; j < c_Rang_mass.Length; j++)
                        {
                            if (str_temp[1] == c_Rang_mass[j])
                            {
                                i_index_v_kolodi_kart_1 = j;
                                break;
                            }
                        }
                    }
```

```
                    else
                    {
                        str_temp += str_cards[i_index_v_kolodi_kart_2_para - 1];//Додаємо в тимчасову змінну карту, яка має змінну, для того порахувати силу пари
                        str_temp += str_cards[i_index_v_kolodi_kart_2_para];
                        for (int j = 0; j < c_Rang_mass.Length; j++)
                        {
                            if (str_temp[4] == c_Rang_mass[j])
                            {
                                i_index_v_kolodi_kart_2 = j;
                                break;
                            }
                        }
                    }
                }
                i_index_v_kolodi_kart_1_para++;
                i_index_v_kolodi_kart_2_para++;
                i_strength_of_hand = 7800 + i_index_v_kolodi_kart_1 + i_index_v_kolodi_kart_2 + i_index_v_kolodi_kart_3 + i_index_v_kolodi_kart_1_para + i_index_v_kolodi_kart_2_para;
                break;
            }
        case "4 of kind":
            {
                int[] kilkist_vhodjen_v_best_combination = new int[5];//Даний масив буде містити кількість входжень елементів в стрічці з найкращою комбінацією
                for (int i = 1, k = 0; k < kilkist_vhodjen_v_best_combination.Length; i += 3, k++)
                    kilkist_vhodjen_v_best_combination[k] = HowOccurrences(str_cards, str_cards[i]);
                for (int i = 0; i < kilkist_vhodjen_v_best_combination.Length; i++)
                {
                    if (kilkist_vhodjen_v_best_combination[i] == 4)
                    {
                        i_index_v_kolodi_kart = i;
                        break;
                    }
                }
                i_index_v_kolodi_kart *= 3;//Так як крок складає 3, то множимо на 3
                str_temp += str_cards[i_index_v_kolodi_kart];//Додаємо в тимчасову змінну карту, яка має змінну, для того порахувати силу пари
                str_temp += str_cards[i_index_v_kolodi_kart + 1];
                for (int i = 0; i < 1; i++)
                {
                    for (int j = 0; j < c_Rang_mass.Length; j++)
                    {
                        if (str_temp[1] == c_Rang_mass[j])
                        {
                            i_index_v_kolodi_kart = j;
                            break;
                        }
                    }
                }
                i_index_v_kolodi_kart++;//Інкрементуємо, тому що масив в C# починається з нуля
                i_strength_of_hand = 9100 + 4 * i_index_v_kolodi_kart + Strength_of_hand_kiker(Hash_Table_Drop);
                break;
            }
        case "Straight flush":
            {
                str_temp += str_cards[0];
                str_temp += str_cards[1];
                for (int i = 0; i < 1; i++)
                {
                    for (int j = 0; j < c_Rang_mass.Length; j++)
                    {
                        if (str_temp[1] == c_Rang_mass[j])
                        {
                            i_index_v_kolodi_kart = j;
                            break;
                        }
                    }
                }
```

```
                i_index_v_kolodi_kart++;//Інкрементуємо, тому що масив в C# починається з нуля
                i_strength_of_hand = 10400 + i_index_v_kolodi_kart + Strength_of_hand_kiker(Hash_Table_Drop);
                break;
            }
        case "Royal Flush":
            {
                i_strength_of_hand = 117000;
                break;
            }
    }
    return i_strength_of_hand;
}

private int Strength_of_hand_kiker(Hashtable Hash_Table_Drop)
{
    ICollection I_Hash_Table_Drop = Hash_Table_Drop.Keys;
    string str_combination_name = "";//Дана змінна буде містити назву комбінації
    string str_cards = "";//Дана змінна буде містити найкращі 5 карт
    string str_temp = "";//Дана змінна буде тимчасово зберігати карти
    int i_strength_of_hand = 0;//Дана змінна буде містити силу руки в числовому евіваленті
    int i_index_v_kolodi_kart = 0;//Дана змінна буде містити індекс найкращої карти в колоді карт
    foreach (string combination in I_Hash_Table_Drop)
    {
        str_combination_name = Hash_Table_Drop[combination].ToString();//Ця змінна буде містити комбінацію, яка є
        str_cards = combination;//Дана змінна буде містити найкращі 5 карт
    }
    str_temp += str_cards[0];
    str_temp += str_cards[1];
    for (int i = 0; i < 1; i++)
    {
        for (int j = 0; j < c_Rang_mass.Length; j++)
        {
            if (str_temp[1] == c_Rang_mass[j])
            {
                i_index_v_kolodi_kart = j;
                break;
            }
        }
    }
    i_index_v_kolodi_kart++;
    return i_strength_of_hand = i_index_v_kolodi_kart;
}

//Даний метод робить ставку цілою
private int The_whole_rate(int i_call)
{
    try
    {
        if (i_call <= 0 || i_call > stack.Stack_oponent)//Генерує помилкку, якщо ставка не коректна
            throw new StackException("Не можливо поставити таку ставку!!!");
    }
    catch (StackException message)
    {
        MessageBox.Show(message.Message + " " + i_call, "Увага помилка", MessageBoxButtons.OK);
    }
    while (i_call % 10 != 0 && i_call < stack.Stack_oponent)
        i_call++;
    return i_call;
}

}
```

Додаток В (обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ МОДЕЛЮВАННЯ ІГРОВОЇ СИТУАЦІЇ В РОЗПОДІЛЕНИХ МАСШТАБОВАНИХ СИСТЕМАХ

Виконав: студент 2-го курсу,
групи 1КН-23м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
_____ Загнітко В. М.
(прізвище та ініціали)

Керівник: к.т.н., проф. каф. КН
_____ Колесницький О.К.
(прізвище та ініціали)
«05» 11 2024 р.

Вінниця ВНТУ - 2024 рік

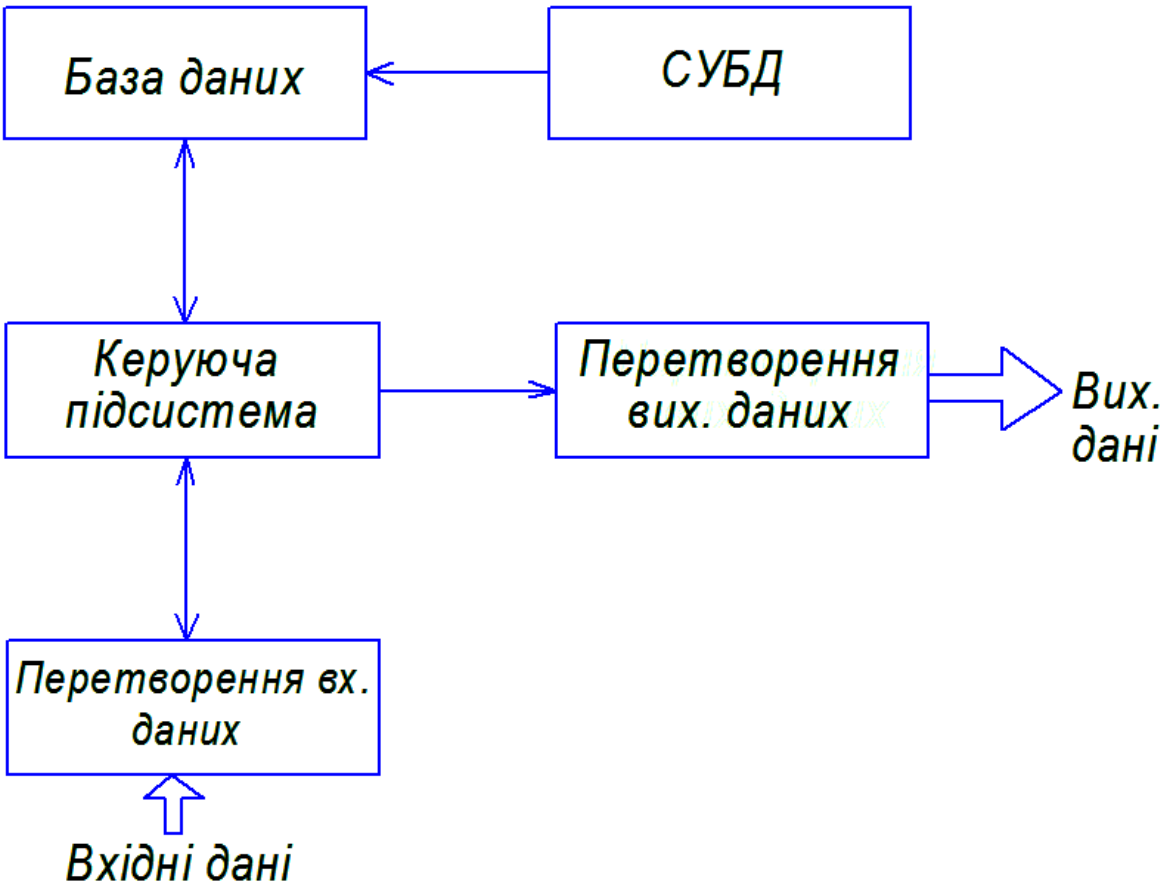


Рисунок В.1 – Структура інформаційної технології моделювання ігрової ситуації «Техаський холдем»

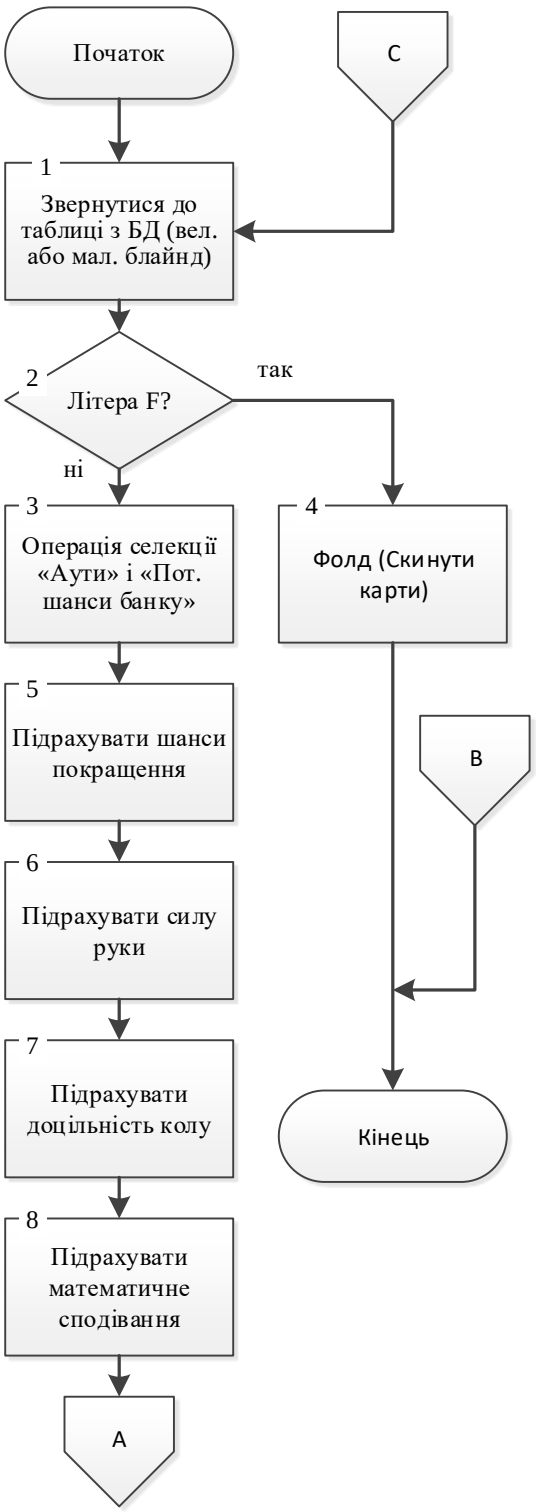


Рисунок В.2 – Граф -схема алгоритму прийняття рішення в «Техаському холдемі»

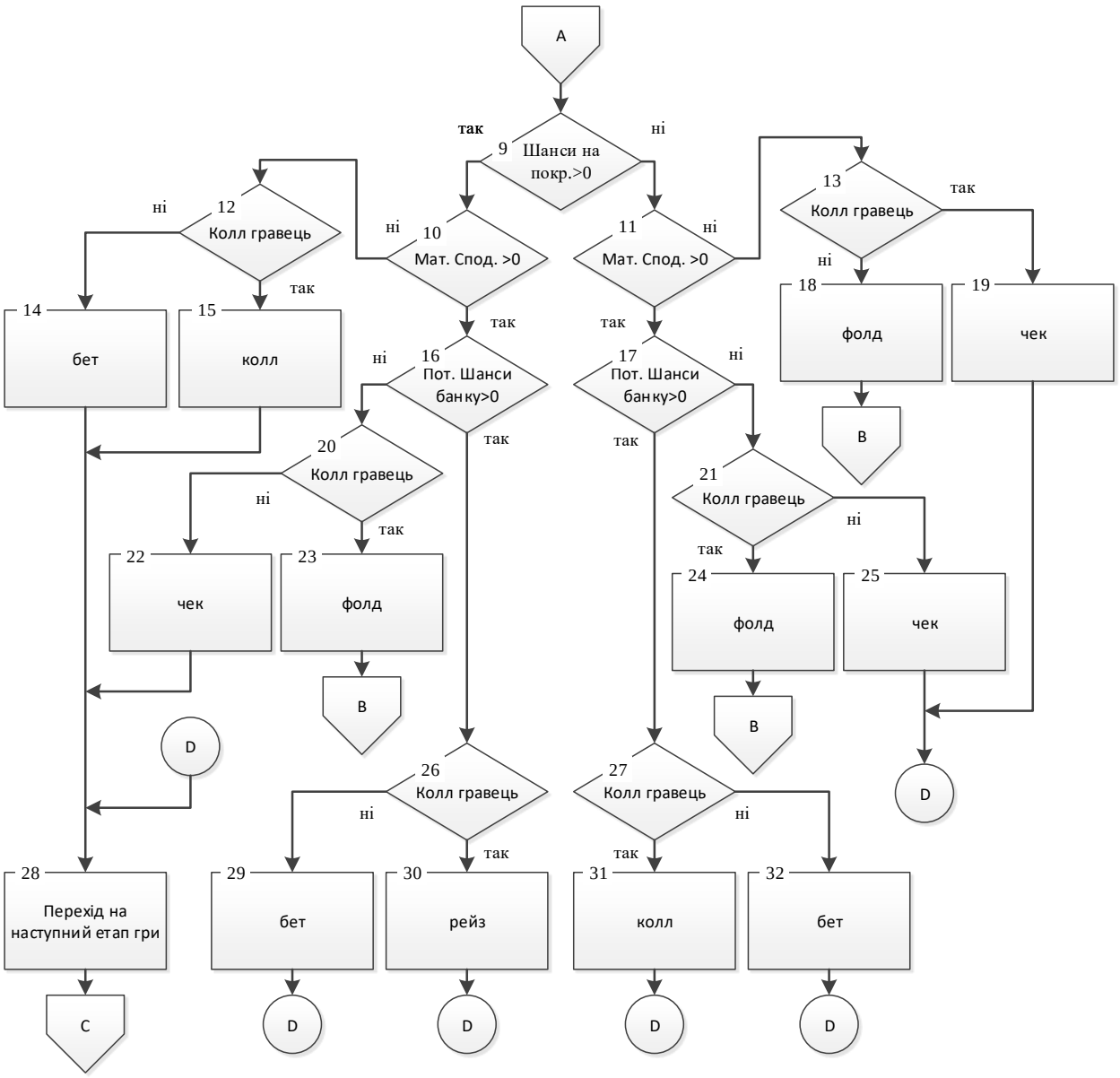


Рисунок В.2 (продовження)

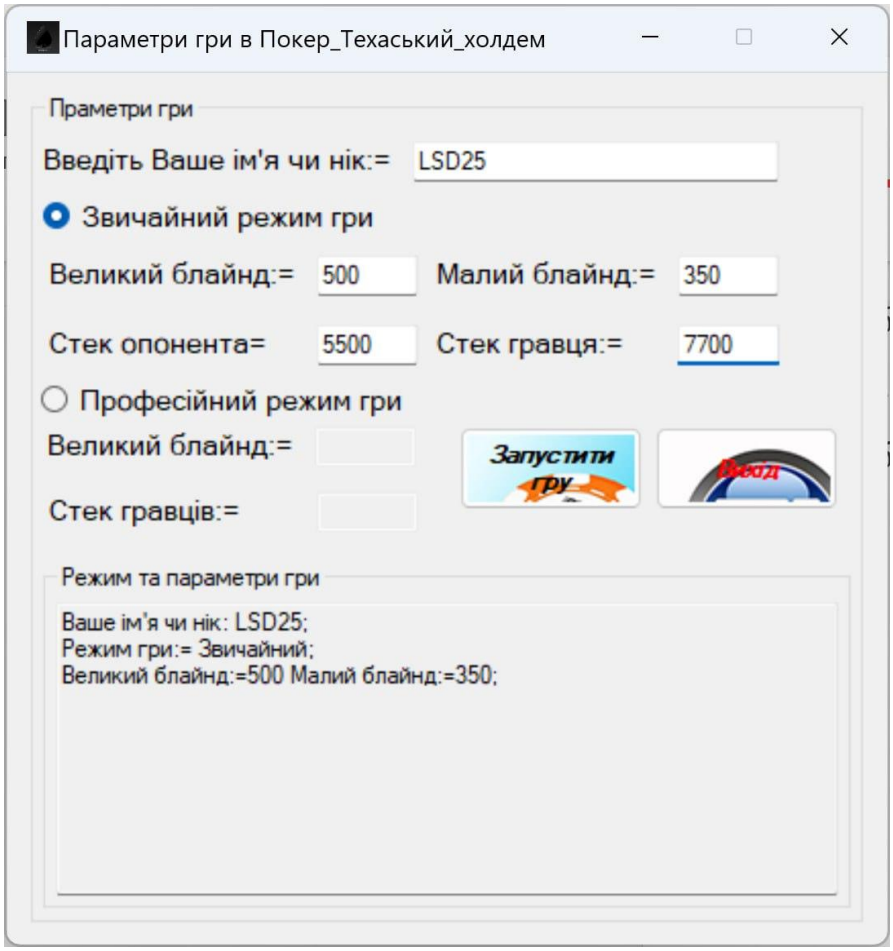
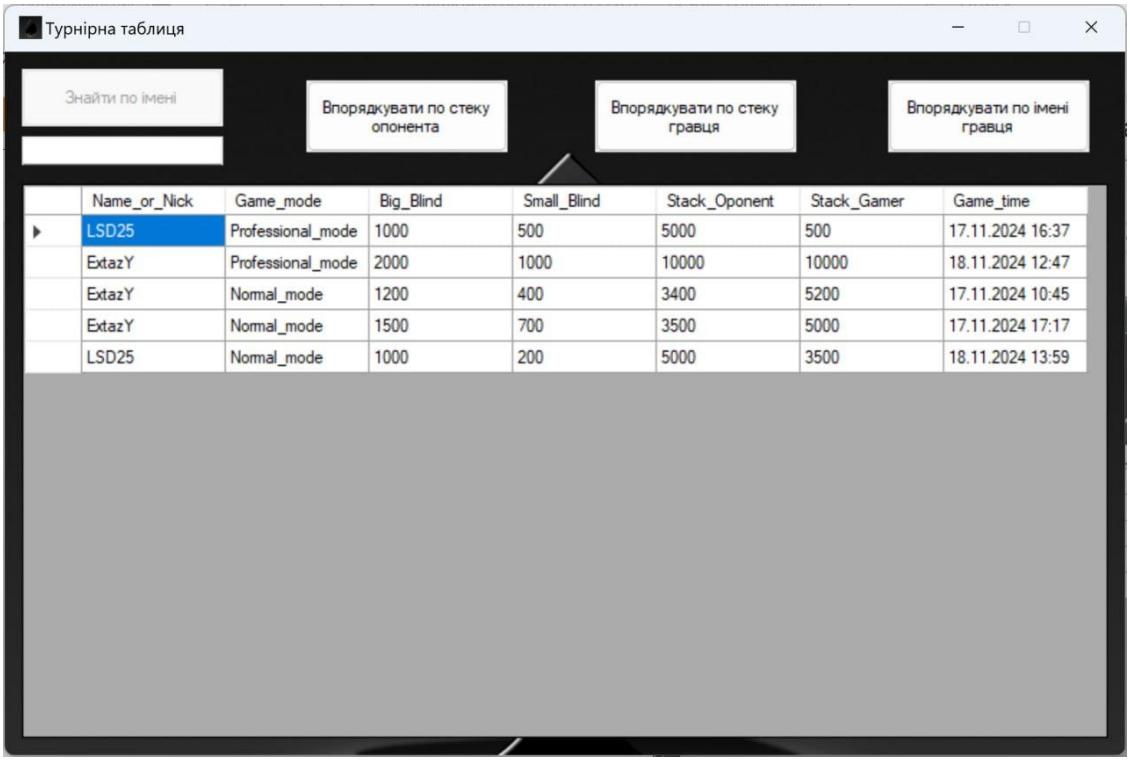


Рисунок В.3 – Головні параметри гри в «Техаському холдемі»

Рисунок В.4 – Стартове вікно та структура робочого поля програми «Техаський холдем»



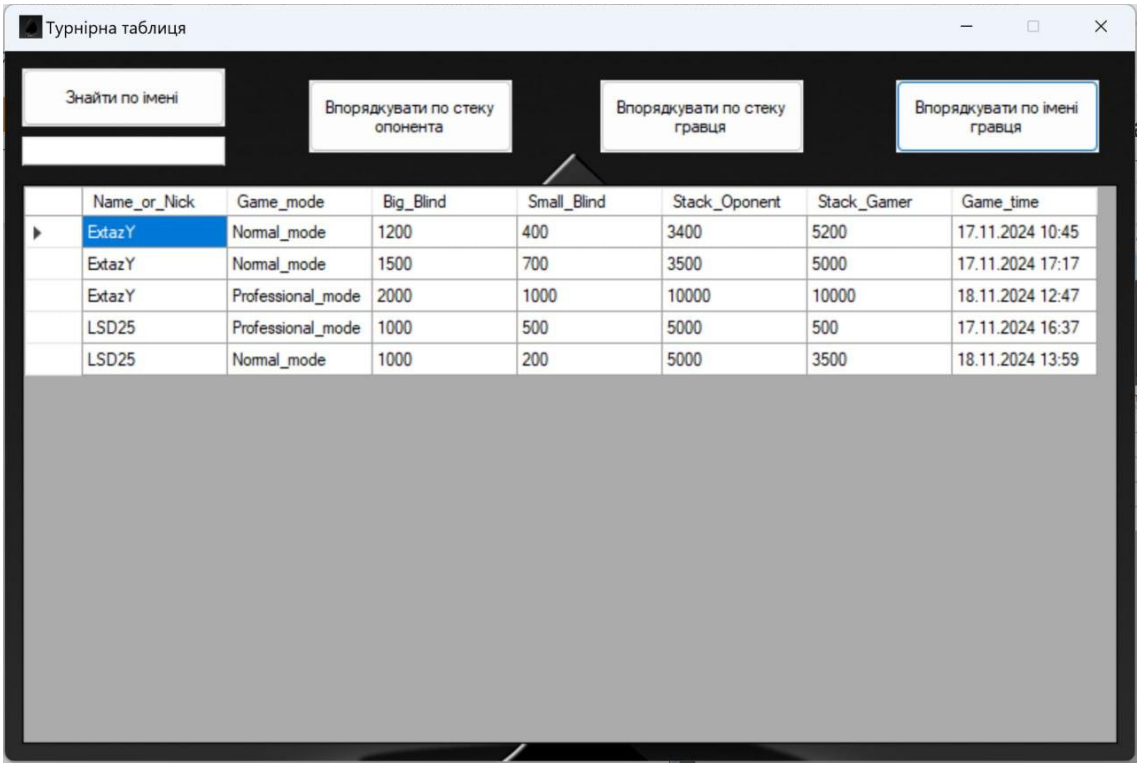


Рисунок В.5 – Приклади відображення турнірних таблиць

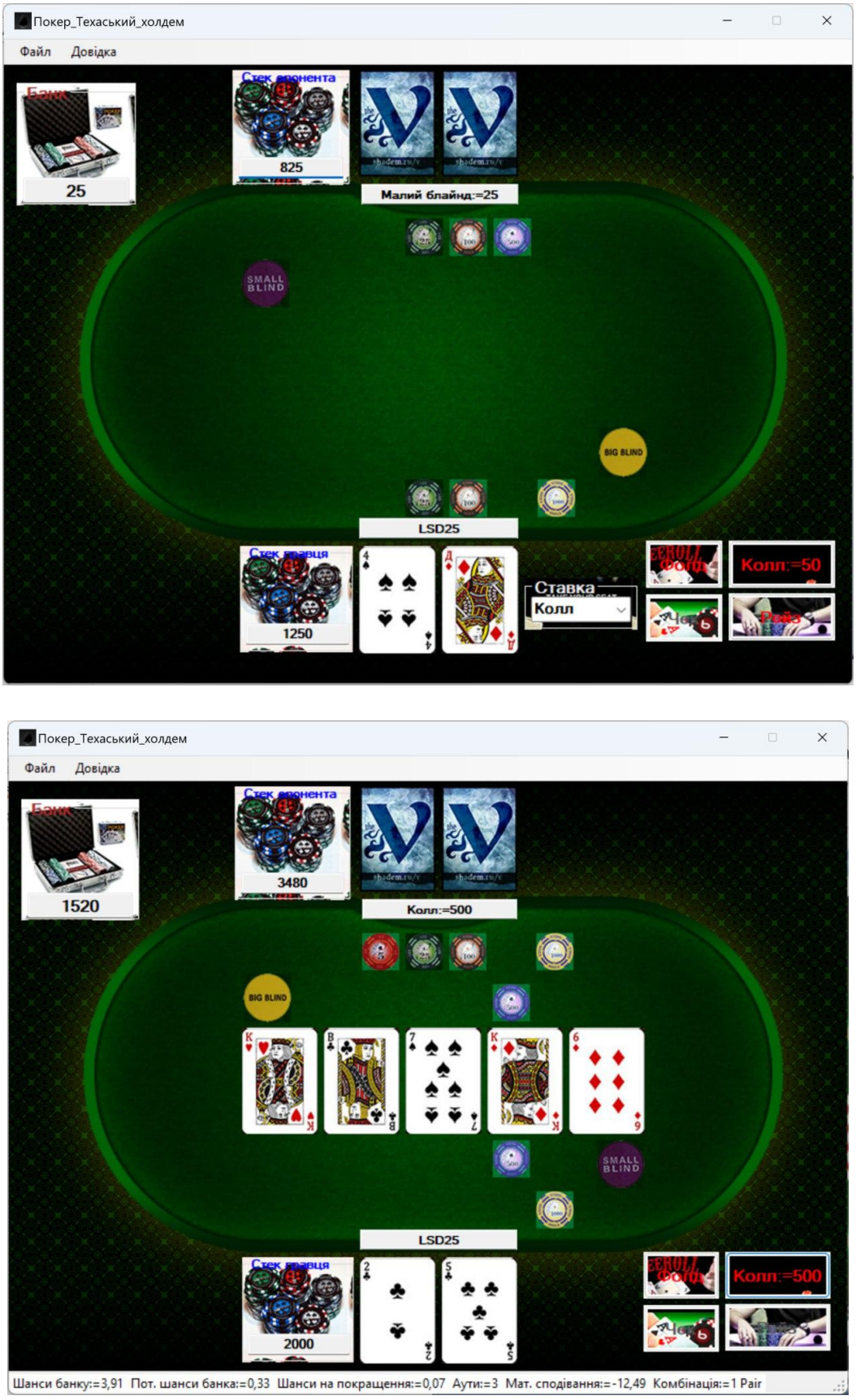


Рисунок В.6 – Приклад результату роботи програми «Техаський холдем»

Додаток Г (довідниковий)

Інструкція користувача

Для запуску гри покер «Техаський холдем» потрібно обов’язково мати встановлену програмну платформу .NET Framework 4.0 та СУБД MS SQL Server 2008 R2. Для запуски можна скористатися файлом «Дипломна робота - Покер Техаський холдем.exe», після запуску даного файлу завантажиться стартове вікно рисунок Г.1.

Рисунок Г.1 – Стартове вікно гри покер «Техаський холдем»

Дана форма (див. рисунок Г.1) містить 3 кнопки: «Запустити гру», «Турнірна таблиця», «Вихід». Кнопка «Турнірна таблиця» завантажує турнірну таблицю гри та дозволяє виконати чотири запити, див. рисунки 4.2 – 4.6. Кнопка

«Вихід» закриває гру, кнопка «Запустити гру» завантажує параметри гри, рисунок Г.2.

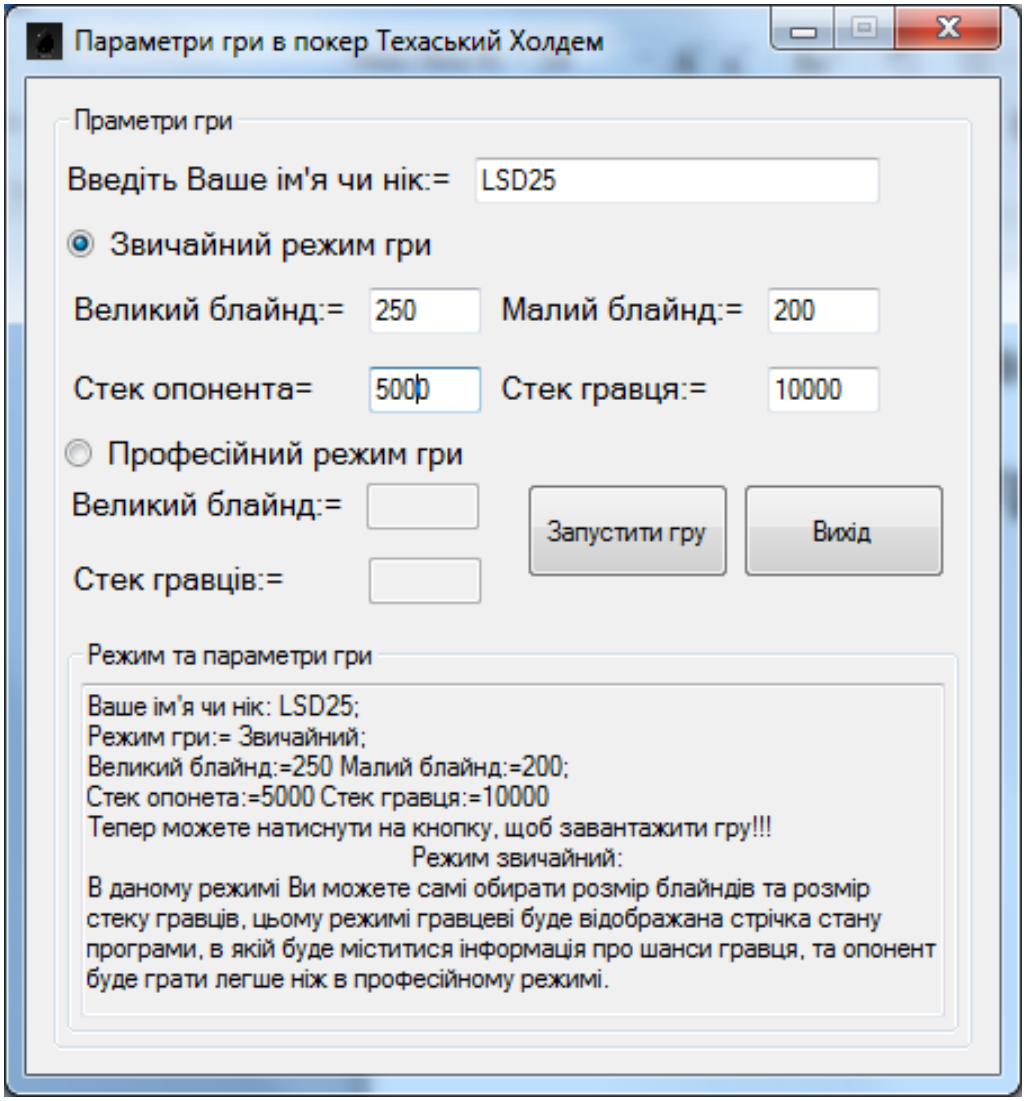


Рисунок Г.2 – Параметри гри покер «Техаський холдем»

В даній формі (див. рисунок Г.2) можна обрати режим гри (звичайний чи професійний), задати блайнди та стек гравців та задати ім’я або нік гравця. В низу форми розташована інформація, щодо обраних параметрів. Форма має 2 кнопки «Запустити гру» та «Вихід», кнопка «Вихід» закриває гру, а кнопка «Запустити гру» завантажує головну форму гри згідно параметрів, які введені в формі «Параметри гри в покер «Техаський холдем» див. рисунок Г.2. На рисунку Г.3 подана головна форма гри.

Рисунок Г.3 – Головна форма гри в покер «Техаський холдем»

В головній формі гри (див. рисунок Г.3) розміщуються руки, фішки та стек обох гравців, в лівому верхньому кутку розміщений банк, куди заносяться ставки та блайнди обох гравців, справа від руки гравця розміщений контролл, який дозволяє обрати ставку гравця, правіша від контролла розміщенні чотири кнопки: «Фолд», «Колл», «Чек» та «Рейз». Ці кнопки використовує користувач для гри в покер «Техаський холдем».

Також для зручності користування грою можна використовувати контекстне меню програми, яке викликається натисканням на праву кнопку мишки. Зверху форми гри розміщене меню програми, яке включає в себе два пункти: «Файл» та «Довідка». Пункт «Файл» має такі підпункти: «Вимкнути нижню стрічку», «Завантажити параметри», «Вихід». пункт «Довідка» має такі підпункти: «Завантажити довідку» та «Про автора».

«Вимкнути нижню стрічку» – вимикає нижню стрічку в формі, за умови, що було обрано в параметрах гри (див. рисунок Г.2) звичайний режим гри.

«Завантажити параметри» – завантажує форму параметрів гри (див. рисунок Г.2).

«Вихід» – завершує гру.

«Завантажити довідку» – завантажує форму, в якій написана довідка до гри в покер «Техаський холдем», рисунок Г.4. Довідка написана на трьох мовах, це дозволить більшій кількості людей читати довідку.

Рисунок Г.4 – Довідка до гри в покер «Техаський холдем»

Додаток Д (довідниковий)

Головні комбінації гри «Техаський холдем»

Флеш-рояль («королівська масть») – старші п’ять карт (туз, король, дама, валет, десять) однієї масті, рисунок Д.1:

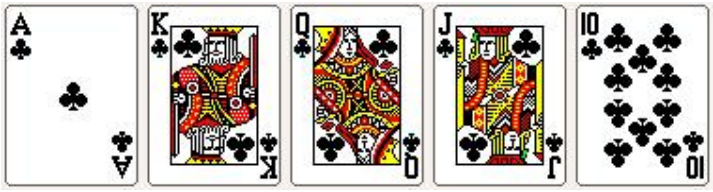


Рисунок Д.1 – Флеш-рояль

Стріт-флеш («масть по порядку») – п’ять карт однієї масті, що йдуть по зростанню без прогалин, рисунок Д.2.

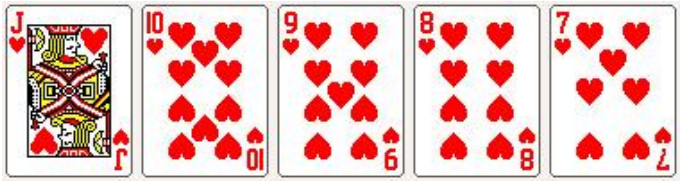


Рисунок Д.2 – Стріт-флеш

Каре («чотири однакових») – чотири карти одного значення, рисунок Д.3.

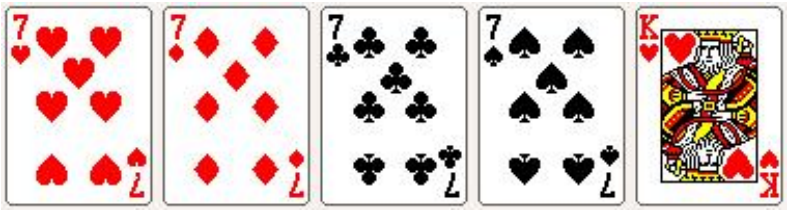


Рисунок Д.3 – Каре

Фул-хаус («повний дім») – три карти одного значення та дві інші карти одного значення, рисунок Д.4.

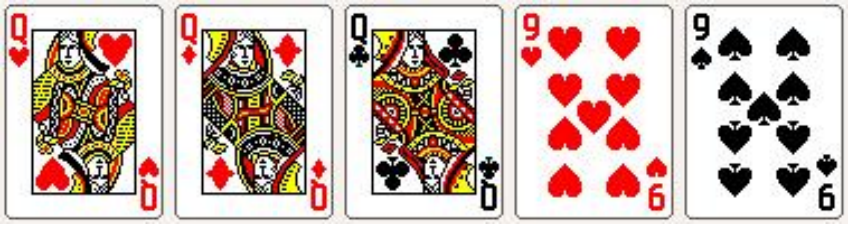


Рисунок Д.4 – Фул-хаус

Флеш («масть») – п’ять карт однієї масті, рисунок Д.5.

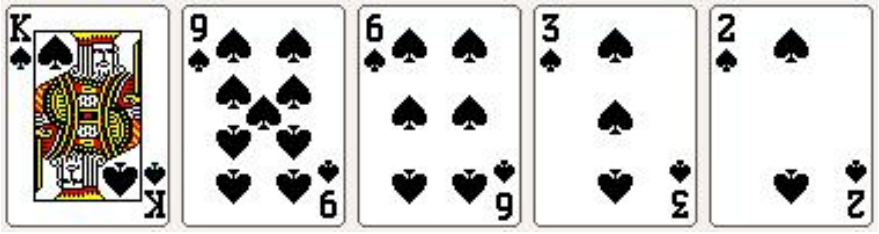


Рисунок Д.5 – Флеш

Стріт («порядок») – карти різних мастей в порядку їх значимості, рисунок Д.6.

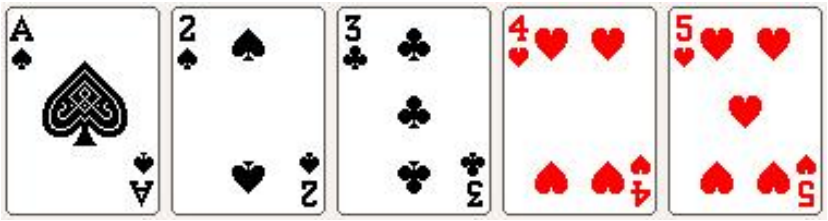


Рисунок Д.6 – Стріт

Сет («три однакових») – три карти одного значення, рисунок Д.7.

Рисунок Д.7 – Сет

Дві пари – дві карти одного значення та дві інші карти одного значення, рисунок Д.8.

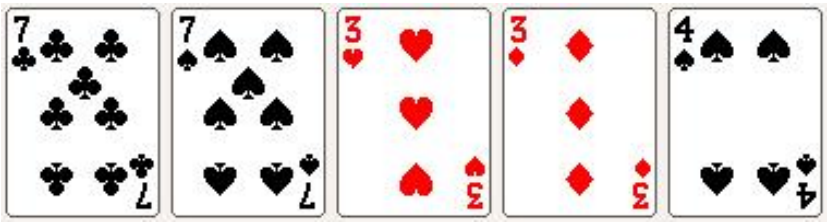


Рисунок Д.8 – Дві пари

Пара – дві карти одного значення, рисунок Д.9.

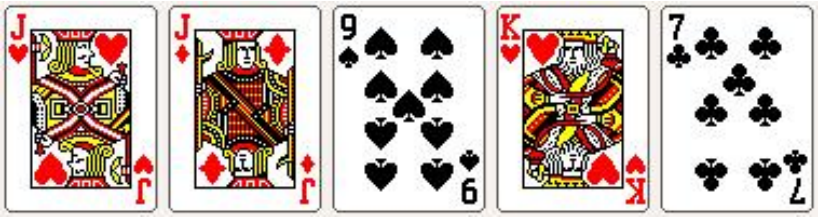


Рисунок Д.9 – Пара

Старша карта(«кікер») – найбільша карта, наприклад (комбінація називається «старший туз»), рисунок Д.10.

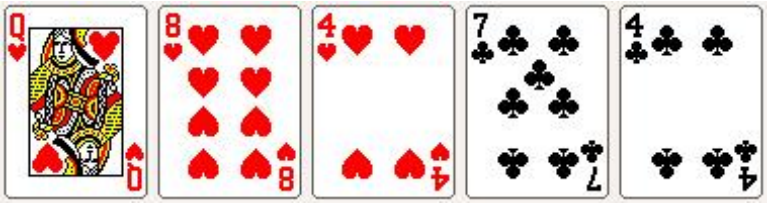


Рисунок Д.10 – Старша карта