УДК 004.94 DOI https://doi.org/10.35546/kntu2078-4481.2025.1.1.15

О. Б. КУНГУРЦЕВ
кандидат технічних наук, професор,
професор кафедри інженерії програмного забезпечення
Національний університет «Одеська політехніка»
ORCID: 0000-0002-3207-7315

П. М. ГУЗАР
студент
Національний університет «Одеська політехніка»
ORCID: 0009-0008-2175-5513

Т. О. НОВІКОВА
асистент кафедри інженерії програмного забезпечення
Національний університет «Одеська політехніка»
ORCID: 0009-0004-5755-9275

Т. О. ФЕДЯНІН
студент
Національний університет «Одеська політехніка»
ORCID: 0009-0008-2175-5513

Б. В. ЧЕБАНЕНКО
студент
Національний університет «Одеська політехніка»
ORCID: 0009-0004-5186-2353

КОМП'ЮТЕРНІ МОДЕЛІ ДЛЯ ГРИ У СНУКЕР І ПІДКАЗОК

Робота спрямована на створення комп'ютерної моделі гри у снукер, яка поряд з імітацією реальної гри проводить аналіз ігрової ситуації та пропонує ефективні способи продовження гри. Показано зростання популярності гри у снукер у всьому світі. Проаналізовано існуючі варіанти комп'ютерних ігор у снукер. Встановлено, що в існуючих моделях гри недостатньо реалізована підтримка гравців-початківців у таких питаннях як посилання на правила в конкретній ситуації, визначення ефективного варіанту продовження гри, вибір точки та напрямку удару по битку тощо. Також немає реалізацій низки варіантів гри, важко визначити ступінь відповідності реальним правилам гри та реальним ситуаціям на ігровому полі. Запропоновано створити комп'ютерну версію гри на базі алгоритмів, що базуються на правилах реальної гри та особливостях, пов'язаних з її комп'ютерною реалізацією. Для підтримки можливих модифікацій програмного забезпечення використано об'єктно-орієнтовану технологію проектування. Створені моделі елементів гри: модель ігрового столу, моделі битка та прицільної кулі, моделі управління грою та управління часом гри. Кожна модель має ряд функцій, які визначають поведінку відповідного об'єкта у певних ситуаціях. Моделі реалізовані у вигляді програмних класів, що дає можливість за необхідності вносити зміни та вдосконалення у програмне забезпечення шляхом наслідування та композиції. Створено програмне забезпечення, яке реалізує запропоновані моделі. В даний час реалізована комп'ютерна гра для версій класичного та швидкого снукера. Розроблено режим підказок, який демонструє гравцю довідку про ситуацію, що виникла, рекомендації щодо вибору удару, дозволяє відновити розміщення куль тощо. Мови спілкування гравця з програмою – українська та англійська. Програмне забезпечення успішно протестоване.

Ключові слова: *комп'ютерна гра, алгоритм, комп'ютерна модель, об'єктно-орієнтований підхід, функція, програмний клас, класичний снукер, швидкий снукер, режим підказок.*

O. B. KUNGURTSEV
Candidate of Technical Sciences, Professor,
Professor at the Department of Software Engineering
Odesa Polytechnic National University
ORCID: 0000-0002-3207-7315

P. M. GUZAR
Student
Odesa Polytechnic National University
ORCID: 0009-0008-2175-5513

T. O. NOVIKOVA
Assistant at the Department of Software Engineering
Odesa Polytechnic National University
ORCID: 0009-0004-5755-9275

T. S. FEDIANIN
Student
Odesa Polytechnic National University
ORCID: 0009-0008-2175-5513

B. V. CHEBANENKO
Student
Odesa Polytechnic National University
ORCID: 0009-0004-5186-2353

COMPUTER MODELS FOR SNOOKER GAME AND TIP

This article focuses on developing a computer model for the game of snooker, which, along with simulating a real game, analyses the game situation and suggests effective strategies for its continuation. The article shows the increasing popularity of snooker all over the world. The article reviews existing computer snooker games and highlights their shortcomings, particularly in supporting novice players. Many current models lack features that clarify rules in specific situations, suggest effective options for the next move, and assist in selecting the point and direction for the cue. Additionally, some game variants are not adequately implemented, and it is challenging to determine how closely they comply with the actual rules of the game and real-life scenarios on the playing field. The article proposes creating a computer version of the game based on algorithms that reflect the real game's rules and the features associated with its implementation. An object-oriented design technology is used to support possible software modifications. Various models of game elements were created, including a model of a game table, cue ball, and target ball, as well as models for game control and time management. Each model has several functions that determine the behaviour of the corresponding object in certain situations. These models are implemented as program classes, allowing for updates and improvements to the software through inheritance and composition when necessary. Software implementing the proposed models has been created. Currently, a computer game for classic and fast snooker versions has been implemented. A hints mode has been developed, providing the player with information about the game situation, recommendations for shot selection, and the ability to restore the placement of balls. The software supports communication in both Ukrainian and English and has been successfully tested.

Key words: *computer game, algorithm, computer model, object-oriented approach, function, program class, classic snooker, fast snooker, hints mode.*

Постановка проблеми

Гра в снукер – один з найбільш поширених і складних різновидів більярду. Снукер дуже популярний у Великій Британії [1], Китаї, набуває все більше прихильників в Азії, західної Європі і на пострадянському просторі [2].

Популярність гри спричинила створення комп'ютерних варіантів гри [3], які стали досить популярні. Однак, практично відсутня інформація про підтримку правил гри, які у комп'ютерному варіанті дещо відрізняються від реальної гри. Крім цього, у відомих іграх мало уваги приділено проблемам, що виникають у недосвідченого гравця: аналіз помилок, демонстрація правильного рішення в певній ситуації, підказка про природу ситуації, що склалася, і застосовувані до неї правила. Таким чином, існує проблема створення моделей гри в снукер із додатковим сервісом для недосвідчених гравців.

Аналіз останніх досліджень і публікацій

У останні роки з'явилася низка досліджень, присвячених фізичним основам гри. У роботі [4] досліджено рух прицільної кулі після зіткнення з битком. Траєкторії руху куль проаналізовано у дослідженні [5]. Визначення траєкторії та швидкості руху кулі після зіткнення з бортом представлено у дослідженні [6]. Повний аналіз руху битка, якому надано різкий імпульс паралельний горизонтальній поверхні, з урахуванням ковзання та обертання наведено у дослідженні [7]. Таким чином фізика руху куль у снукері досить вивчена і може бути реалізована в комп'ютерних моделях гри. Одночасно у відомих варіантах гри [3]. майже відсутні підказки, елементи тренування гравців, аналіз ситуацій. Відсутність комп'ютерних моделей гри не дозволяє зрозуміти повноту реалізації правил гри.

Формулювання мети дослідження

Метою роботи є створення комп'ютерної моделі гри, яка дозволяє реалізувати різні варіанти гри і одночасно забезпечити підтримку недосвідченим гравцям у вигляді підказок і аналізу ситуацій по ходу гри.

Викладання основного матеріалу досліджень

Відповідно до визначеної мети запропоновано створити наступні математичні моделі: модель столу, модель битка, модель прицільної кулі, моделі управління грою та часом гри, модель тренувального режиму.

Модель битка має всю інформацію щодо розташування битка та параметрів удару.

$$CueBall = \langle InitCoord, CurrentCoord, MotionVector, Rotation, Slowdown, CueTilt, ObjectBall \rangle, \quad (1)$$

де $InitCoord = \langle x_0, y_0 \rangle$ – початкові координати кулі; $CurrentCoord = \langle x_t, y_t \rangle$ – поточні координати кулі; $MotionVector = \langle \Delta_x, \Delta_y \rangle$ – вектор руху кулі; $Rotation = \langle x_b, y_b \rangle$ – обертання кулі; $Slowdown$ – уповільнення руху кулі; $CueTilt$ – нахил кия при ударі; $ObjectBall = RedN,N = 1\ldots15 | Yellow | Green | Brown | Blue | Pink | Black$ прицільна куля (для кольорових куль – замовлена куля).

Функції моделі битка представлено у таблиці 1. Якщо функції мають аргументи, то це показано трьома крапками у дужках.

Таблиця 1

Функції моделі битка

№	Назва функції	Призначення функції
1	*fShotCueBall*(…)	Удар по битку
2	*fCollisionCueBall*()	Зіткнення з бортом, губками лузи, іншою кулею
3	*fPlayHand*(…)	Биток грається з руки
4	*fReturnCueBall*()	Фол, пов'язаний з битком

Модель прицільної кулі може представляти червоні і кольорові кулі

$$ObjectBall = \langle TypeBall, InitCoord, CurrentCoord, MotionVector, Slowdown \rangle \quad (2)$$
$$TypeBall = RedN,N = 1..15 | Yellow | Green | Brown | Blue | Pink |,$$

де $InitCoord = \langle x_0, y_0 \rangle$ – початкові координати кулі залежать від типу кулі $InitCoord = f(TypeBall)$. Для червоної кулі це координати у межах трикутника, які використовуються тільки на початку партії. Для кольорової кулі це певна позиція на столі, яка використовується багато разів у процесі гри. Крім того, якщо куля потрапляє у лузу, то саме *TypeBall* визначає кількість очок.

Таблиця 2

Функції моделі прицільної кулі

№	Назва функції	Призначення функції
1	*fRBinPocket*(…)	Червона куля потрапляє у лузу
2	*fCBinPocket*(…)	Кольорова куля потрапляє у лузу
3	*fScoreChange*(..)	Нарахування очок гравцю

Функція *fScoreChange*(..) визначає очки, які нараховані гравцям відповідно до типу кулі (звичайні або штрафні).

Модель столу включає ігрове поле і лузи (рис. 1), моделі куль і виконує основні дії щодо визначення зіткнень і повернення куль.

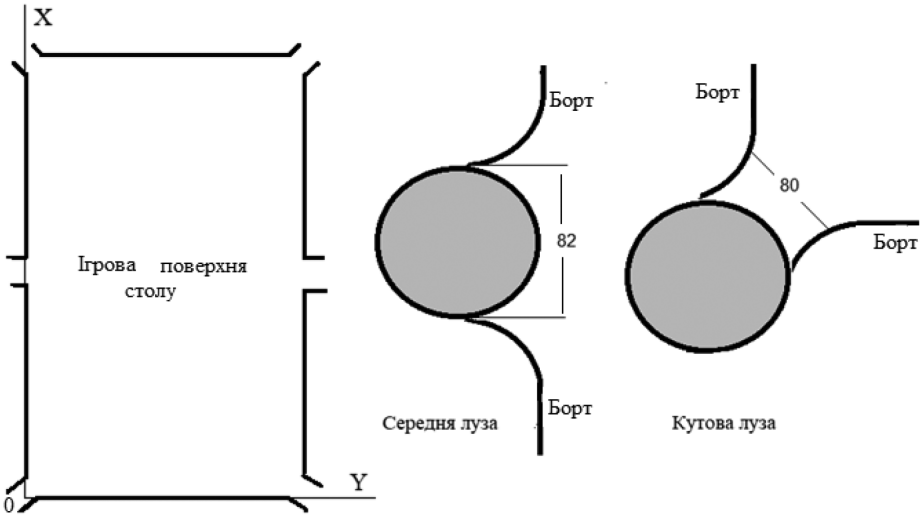

Рис. 1. Геометрія столу і луз

$$SnookerTable = \langle Table, mBallPosition, mBallCurrentPosition \rangle \quad (3)$$

$Table = \langle PlayingField, mSide, mPocket \rangle$ – стіл;

PlayingField = ⟨*Length,Width*⟩ – ігрове поле. Стандартна довжина – *Length* = 36 576 мм, стандартна ширина – *Width* = 18 288 мм.

mSide = {*TopSide,BottomSide,LeftSide,RightSide*} – борти;

mPocket = {*TopLeftPocket, TopRightPocket, MiddleLeftPocket, MiddleRightPocket, BottomLeftPocket, BottomRightPocket*} – множина луз.

mBallPosition = {⟨*BallName,X0Ball,Y0Ball*⟩} – початкове розташування куль, включає назву кулі – *BallName* і координати на столі – *X0Ball,Y0Ball*;

mBallCurrentPosition = {⟨*BallName, XcBall, YcBall, OnTable*⟩} – поточне розташування куль. Якщо куля відсутня, її координати будуть від'ємні.

Таблиця 3

Функції моделі столу

№	Назва функції	Призначення функції
1	*fInitPlaceBalls*()	Розстановка куль для початку гри
2	*fBallsCollision*(…)	Зіткнення двох куль
3	*fBallCollidedSide*(…)	Зіткнення кулі з бортом
4	*fBallCollidedPocketJaws*(…)	Зіткнення кулі з губками лузи
5	*fPocketingRB*()	Влучення в лузу червоної кулі
6	*fPocketingCB*(…)	Влучення в лузу кольорової кулі
7	*fReturnCB*(..)	Повернення до столу кольорової кулі
8	*fNoCollisionBallAndSide*()	Відсутність торкання борта кулею

Функція *fBallsCollision*(…) розраховує вектори руху двох куль після зіткнення на основі векторів їх руху перед зіткненням.

Функція *fBallCollidedPocketJaws*(…) розраховує вектор руху кулі в залежності від точки зіткнення з губкою на основі вектору руху кулі перед зіткненням.

Модель керування грою потрібна для визначення ігрових ситуацій, нарахування очок, виконання певних дій у тренувальному режимі тощо.

$$GameControl = \langle Players,APlayer,Points,Winner,MaxFr,SnookerSituation\rangle, \quad (4)$$

де *Players* = ⟨*Player1, Player2*⟩ – перший і другий гравці; *Player_i* = ⟨*FirstName,LastName,Country,Rating*⟩; *APlayer* – активний гравець; *Points* = ⟨*Points1,Points2,FramesWon1,FramesWon2*⟩ – кількість очок першого і другого гравця у поточному фреймі; кількість виграних фреймів першим і другим гравцем; *Winner* – переможець фрейму; *MaxFr* – максимальна кількість фреймів у зустрічі; *SnookerSituation* – ситуації, які вимагають застосування певних правил: "*Foul*", "*Foul and Miss*", "*Snooker*", "*Free ball*".

Таблиця 4

Функції моделі керування грою

№	Назва функції	Призначення функції
1	*fActivePlayer*()	Визначення активного гравця
2	*fDefSituation*()	Визначення певної ситуації під час гри
3	*fAddingPoints*()	Додавання очок гравцю
4	*fDefWinner*()	Визначення переможця
5	*fUpdateStatistics*()	Оновити статистику по гравцям і турніру
6	*fFixCenturyBr*()	Фіксація брейку в 100 очок
8	*fRepeatSituation*(…)	Повторення ситуації
9	*fDemoCorrectShot*(…)	Демонстрація «правильного» удару
10	*fObjectBallSelection*(…)	Вибір прицільної кулі
11	*fDemoSituation*()	Демонстрація інформації відносно ситуації

Функція *fRepeatSituation*(…) дозволяє повторювати ситуацію для прийняття гравцем кращого рішення. Функція *fDemoCorrectShot*(…) демонструє «правильну» точку удару і зіткнення, а також напрям удару для придання певного напрямку руху прицільної кулі (рис. 2). Функція *fDemoSituation*() дає розгорнуту інформацію щодо ігрової ситуації. Функція *fObjectBallSelection*(…) дозволяє обрати прицільну кулю з найбільшою ймовірністю влучення у лузу.

Модель керування часом гри реалізує варіант гри – швидкий снукер, передбачає обмеження часу на підготовку удару і деякі інші зміни у правилах гри.

$$GameTimeManagement = \langle TTime,CTime,STime,APTime\rangle, \quad (5)$$

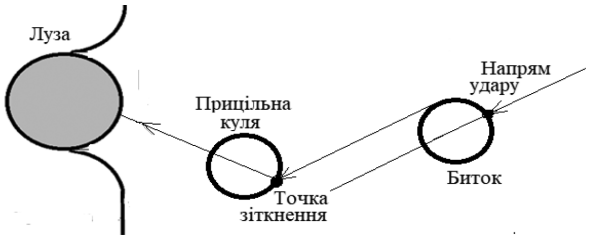

Рис. 2. Демонстрація «правильного» удару в тренувальному режимі

Таблиця 5

Функції моделі керування часом гри

№	Назва функції	Призначення функції
1	*fDataInit*()	Ініціалізація даних
2	*EndOfTimeWarning*()	Попередження щодо кінця часу
3	*fDefFoul*()	Визначення фолу
4	*fSetShotPrepareTime*()	Встановлення часу підготовки удару

де *TTime* – ліміт часу на фрейм; *CTime* – поточний час фрейму; *STime* – ліміт часу на підготовку удару; *APTime* – поточний час гравця під час підготовки до удару.

Модель тренувального режиму дає можливість відтворювати ігрові ситуації, фрагменти гри та виконувати підказки.

$$TrainingMode = \langle mSituation, mGamePieces \rangle, \quad (6)$$

де *mSituation* – множина ігрових ситуацій; *mGamePieces* – множина фрагментів гри.

Таблиця 6

Функції моделі тренувального режиму

№	Назва функції	Призначення функції
1	SaveGamePiece(…)	Збереження фрагменту гри
2	SaveSituation(…)	Збереження ігрової ситуації
3	ReplicateSituation(…)	Демонстрація ігрової ситуації
4	ReplicateGamePiece(…)	Демонстрація фрагменту гри
5	DemoCorrectShot(…)	Демонстрація коректного удару
6	ObjectBallSelect(…)	Вибір прицільної кулі

Діаграма класів представляє важливий етап створення програмного забезпечення для гри. Розглянуті моделі дають змогу реалізувати об'єктно-орієнтований підхід до розробки програми гри [8]. Це дозволяє реалізувати відносини спадкування та агрегації [9], що спрощує модифікацію програми.

На рис. 3 представлені програмні класи, що реалізують варіант класичного снукера, швидкого снукера (Shot out) та режим тренування.

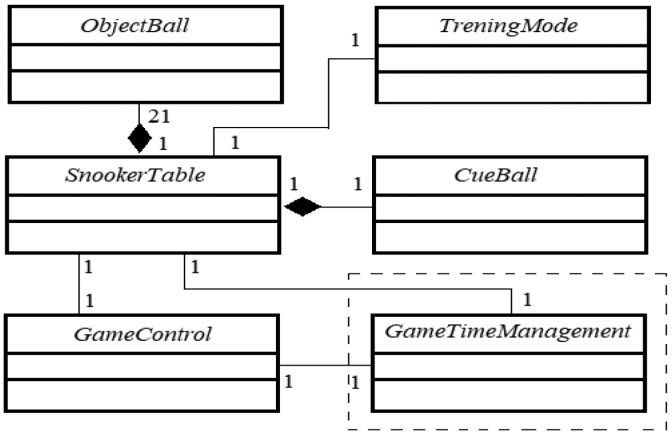

Рис. 3. Діаграма програмних класів

На основі розроблених моделей створено програму ESnookerCoach, яка успішно пройшла тестування.

На рис. 4 показано типовий момент тренування і підказку програми щодо вибору найкращої прицільної кулі (функція ObjectBallSelect(…)).

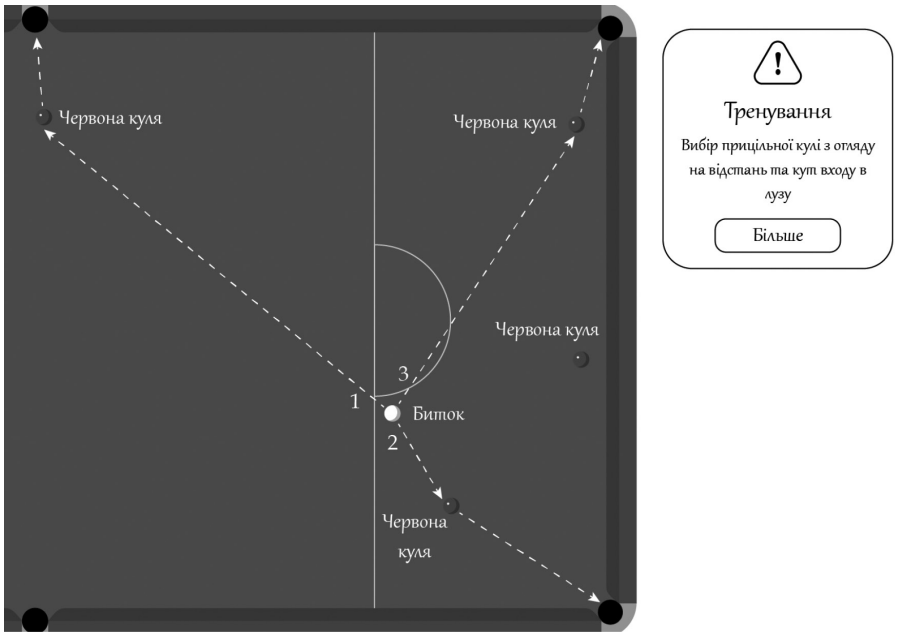

Рис. 4. Підказка щодо вибору прицільної кулі

На рис. 5 показано момент гри, коли зафіксовано ситуацію «Снукер» (функція *fDefSituation*()) і виведено додаткову інформацію (функція *fDemoSituation*()).

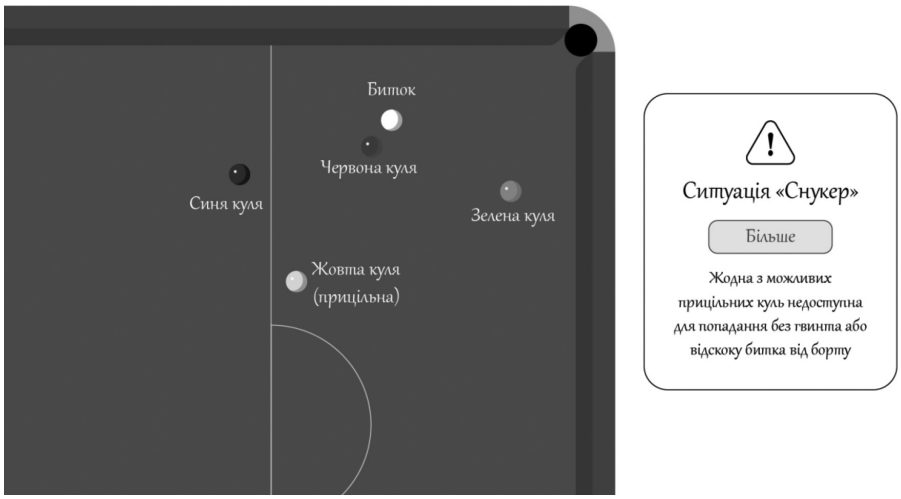

Рис. 5. Повідомлення щодо ситуації, що виникла

Висновки

Встановлено необхідність розробки моделей комп'ютерної гри в снукер для гравців-початківців. Створено моделі основних компонентів гри: столу, битку, прицільної кулі, організації гри, керуванням часом, тренувального режиму. Розроблені функції, що дозволяють швидше зрозуміти і освоїти гру для гравців-початківців. Створена програма реалізує гру на основі об'єктно-орієнтованої технології, що спрощує подальше вдосконалення моделі. Подальші дослідження спрямовані на моделювання ситуацій з бажаним виходом битка після удару та ефективним варіантом відіграшу.

Список використаної літератури

1. Rhys-Taylor, A. Changing Britain through the frame of snooker // *Sport in Society*. 2024, 1–17. URL: https://doi.org/10.1080/17430437.2024.2389812
2. Harris, L. J. The global snooker market // *Sport in Society*. 2021. № 24(10). Pp. 1718–1728. URL: https://doi.org/10.1080/17430437.2021.1901341
3. Steve C. Snooker 19 Review. (дата звернення 30.04.2019) URL: https://www.thesixthaxis.com/2019/04/30/snooker-19-review/
4. Hyeong-Chan Kim. Motions of a billiard ball after a cue stroke // New Phesics: Sae Mulli. 2022. Vol. 72, No. 3. Pp. 208–223. URL: http://dx.doi.org/10.3938/NPSM.72.208
5. Grzegorz Kudra, Michał Szewc, Igor Wojtunik, Jan Awrejcewicz. Shaping the trajectory of the billiard ball with approximations of the resultant contact forces// Mechatronics. 2016. Volume 37. Pp 54-62. URL: https://doi.org/10.1016/j.mechatronics.2016.01.002
6. Antonio Doménech-Carbó. Independent friction-restitution description of billiard ball collisions//Mechanics Research Communications. 2023. Volume 131, 104149. URL: https://doi.org/10.1016/j.mechrescom.2023.104149
7. Abdullah Ozkanlar. Billiard Physics: Motion of a Cue Ball after Hit by a Cue Stick Horizontally // The Physics Educator. 2020. Vol. 02, No. 01, 2050003. URL: https://doi.org/10.1142/S2661339520500031
8. Kungurtsev O.B., Novikova N.O., Zinovatna S.L., Komleva N.O //Automated object-oriented for software module development. Applied Aspects of Information Technology. 2021. Vol. 4 No. 4. Pp. 338–353. URL: https://doi.org/10.15276/aait.04.2021.4
9. Kungurtsev, O., & Komleva, N. Implementation of class interaction under aggregation conditions//Eastern-European Journal of Enterprise Technologies. 2024. 2/2 (128). Pp. 20–30. URL: https://doi.org/10.15587/1729-4061.2024.301011

References

1. Rhys-Taylor, A. Changing Britain through the frame of snooker // *Sport in Society*. 2024, 1–17. URL: https://doi.org/10.1080/17430437.2024.2389812
2. Harris, L. J. The global snooker market // *Sport in Society*. 2021. № 24(10). Pp. 1718–1728. URL: https://doi.org/10.1080/17430437.2021.1901341
3. Steve C. Snooker 19 Review. (дата звернення 30.04.2019) URL: https://www.thesixthaxis.com/2019/04/30/snooker-19-review/
4. Hyeong-Chan Kim. Motions of a billiard ball after a cue stroke// New Phesics: Sae Mulli. 2022. Vol. 72, No. 3. Pp. 208–223. URL: http://dx.doi.org/10.3938/NPSM.72.208
5. Grzegorz Kudra, Michał Szewc, Igor Wojtunik, Jan Awrejcewicz. Shaping the trajectory of the billiard ball with approximations of the resultant contact forces// Mechatronics. 2016. Volume 37. Pp 54–62. URL: https://doi.org/10.1016/j.mechatronics.2016.01.002
6. Antonio Doménech-Carbó. Independent friction-restitution description of billiard ball collisions //Mechanics Research Communications. 2023. Volume 131, 104149. URL: https://doi.org/10.1016/j.mechrescom.2023.104149
7. Abdullah Ozkanlar. Billiard Physics: Motion of a Cue Ball after Hit by a Cue Stick Horizontally // The Physics Educator. 2020. Vol. 02, No. 01, 2050003. URL: https://doi.org/10.1142/S2661339520500031
8. Kungurtsev O.B., Novikova N.O., Zinovatna S.L., Komleva N.O //Automated object-oriented for software module development. Applied Aspects of Information Technology. 2021. Vol. 4 No. 4. Pp. 338–353. URL: https://doi.org/10.15276/aait.04.2021.4
9. Kungurtsev, O., & Komleva, N. Implementation of class interaction under aggregation conditions//Eastern-European Journal of Enterprise Technologies. 2024. 2/2 (128). Pp. 20-30. URL: https://doi.org/10.15587/1729-4061.2024.301011