DOI: https://doi.org/10.36910/6775-2524-0560-2021-42-22
УДК 004.4`24
Кукол Антон Євгенійович, студент
Хомяк Марія Ярославівна, канд. фіз.-мат. наук, доцент
https://orcid.org/0000-0002-9245-7993
Гришанович Тетяна Олександрівна, канд. фіз.-мат. наук, старший викладач
https://orcid.org/0000-0002-3595-6964
Волинський національний університет імені Лесі Українки, м. Луцьк, Україна

РОЗРОБКА ІГРОВОЇ ПРОГРАМИ «СУДОКУ» ЗА ДОПОМОГОЮ СЕРЕДОВИЩА ПРОГРАМУВАННЯ PYCHARM

Кукол А. Є., Хомяк М.Я., Гришанович Т. О. Розробка ігрової програми «Судоку» за допомогою середовища програмування PyCharm. У роботі досліджено засоби розробки програмного забезпечення для створення гри «Судоку». Проаналізовано існуючі алгоритми та програмні засоби гри «Судоку» та запропоновано власний аналог гри, яка розроблена на мові програмування Python в середовищі PyCharm.
Ключові слова: Судоку, розробка гри, мова програмування Python, середовище PyCharm

Кукол А. Є., Хомяк М.Я., Гришанович Т. А. Разработка игровой программы «Судоку» с помощью среды программирования PyCharm. В работе исследованы средства разработки программного обеспечения для создания игры «Судоку». Проанализированы существующие алгоритмы и программные средства игры «Судоку» и предложено собственный аналог игры, которая разработана на языке программирования Python в среде PyCharm.
Ключевые слова: Судоку, разработка игры, язык программирования Python, среда PyCharm

Kukol A. Ye., Khomyak M.Ya. Hryshanovych T. O. Development of game sudoku by using the PyCharm integrated development environment. The paper considers the software development tools for creating game Sudoku. The existing algorithms and software of game Sudoku were analyzed and the analogue game was developed. The game is developed in the Python programming language in the PyCharm environment.
Keywords: Sudoku, game development, Python programming language, PyCharm IDE.

Постановка проблеми. Однією з популярних головоломок на сьогоднішній день є гра «Судоку». Головоломка «Судоку» допомагає розвинути увагу, логічне мислення, тренує пам'ять. Її суть достатньо проста, потрібно заповнити ігрове поле цифрами, в класичному випадку, від 1 до 9. Особливість полягає в тому, що числа в рядках, стовпцях та дев'ятьох квадратах розміром 3×3 не повинні повторюватись. З самого початку, в залежності від вибору рівня складності, ігрове поле заповнене певною кількістю чисел, що розміщені у клітинках випадковим чином. В результаті має утворитись повністю заповнена таблиця з числами. Судоку зазвичай має один розв'язок, проте існують і спеціальні судоку, в яких кількість розв'язків є відмінною від одиниці. На даний момент існує декілька видів цієї гри. Класичним варіантом судоку вважають таблицю розміром 9×9. Проте, для досвідчених гравців існує гра розміром 15×15, та, навіть 16×16. Та розмір – не єдина характеристика, за якою поділяють судоку. Існують ще деякі види судоку: Віндоку, Латиниця, Жирандоль та ін.

Аналіз актуальних досліджень. На даний момент великою популярністю користується онлайн версія гри «Судоку» Sudoku.com.

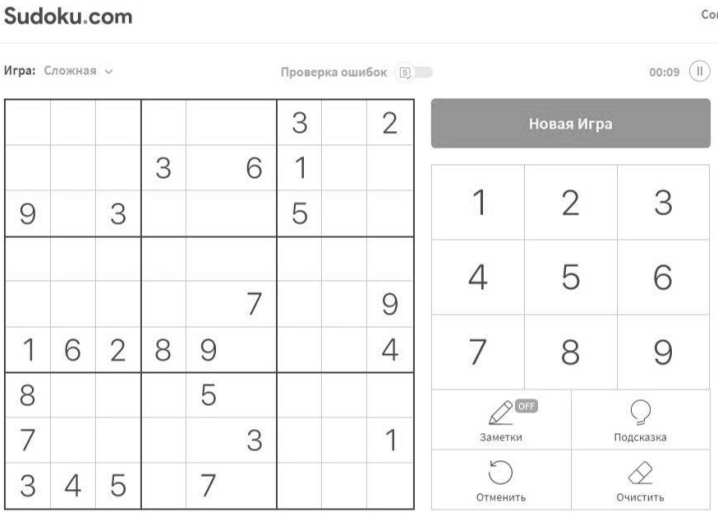


Рисунок 1 - Інтерфейс Sudoku.com

Оригінальність цього продукту полягає у тому, що існує можливість здійснити перевірку помилок, скористатись підказкою, що покаже, яке з чисел користувач поставив неправильно або ж, яке число потрібно ставити в виділену клітинку. Цікавою функцією є кнопка примітки, що дозволяє в клітинці записувати всі можливі варіанти відповіді. Окрім, введення з клавіатури, присутня також можливість вибору чисел з спеціального блоку, де є всі числа від 1 до 9. Це дає змогу керувати лише мишею, без використання клавіатури. Також є кнопка відміни дії.

Мобільна версія гри розроблена компанією Brainuim Studios. ЇЇ перевагами є зручний інтерфейс та система вивчення правил судоку на прикладах.

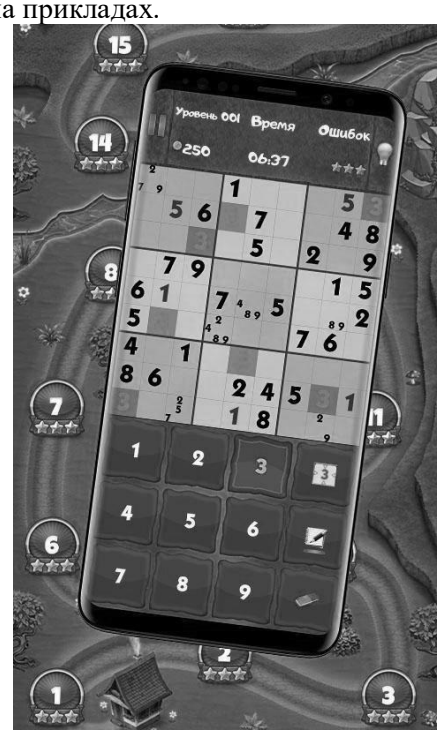


Рисунок 2 - Інтерфейс гри «Sudoku»

В мобільному застосунку присутні такі можливості: таблиця рекордів, досягнення, посібник з відповідями, підказки. При натисканні на клітинку, вона виділяється і з'являються дві лінії, горизонтальна та вертикальна, що допомагає візуально краще оцінювати рядки та стовпці, а квадрат (3x3) в якому знаходиться ця клітинка, зафарбовується однаковим кольором.

В комп'ютерних версіях гри, інтерфейс ігрового та допоміжних полів ненав'язливий та простий. Це зроблено для того, щоб не відволікати гравця від розв'язання поставленої задачі і сконцентрувати його увагу. Щодо мобільних версій, то вони являють собою кольорові варіанти, проте поєднання кольорів дуже вдале і не відволікає користувачів від головоломки.

Метою роботи є дослідження процесу та засобів розробки ігрової програми «Судоку». Зокрема, програмний продукт повинен мати архітектуру desktop-application. Ігрова програма повинна генерувати ігрове поле розмірності 9 на 9, дозволяти гравцеві вводити цифри. У випадку введення невірної цифри або перемоги у грі відповідні повідомлення повинні бути виведені у консоль.

Виклад основного матеріалу. Алгоритм, за яким працює ігрова програма "Судоку" виглядає наступним чином:

1) створюється ігрове поле, а саме таблиця розміром 9x9, без заданих значень;
2) користувачеві надається можливість ввести лише цифри, можливість літер або символів виключена. У випадку такого введення генерується повідомлення про помилку;
3) якщо користувач вирішив самостійно заповнити ігрове поле значеннями, він може самостійно вручну подвійним натиском миші задати їх з клавіатури і розпочати гру в будь-який момент;
4) при кожній зміні чисел на ігровому полі, програма зчитує дані, що були введені, та виконує перевірку на наявність значень, що вже повторюються в клітинках рядків, стовпців та так званих вузлах (квадратах 3x3);
5) якщо гравець припускається помилки, він отримає повідомлення від програми;
6) якщо користувач заповнив ігрове поле без повторень в рядках, стовпцях та вузлах – він отримає повідомлення про перемогу;

Для розробки ігрової програми було використано засоби мови програмування Python. В якості середовища розробки було вибрано крос-платформне інтегроване середовище JetBrains PyCharm, що є

безкоштовним для освітніх начальних закладів та проєктів із відкритим програмним кодом (перебуває під ліцензією Apache License) [7]. PyCharm розроблений на основі платформи IntelliJ IDEA. Користувачам надається можливість самостійно розширювати можливості PyCharm за допомогою написанння плагінів. Деякі із плагінів для інтегрованих середовищ розробки, що розробляються JetBrains, є сумісними із PyCharm.

Для створення гри у середовищі PyCharm було створено проєкт, який складається з наступних файлів:

GUI.py – це проєктний файл, в якому необхідно підключити вихідний файл (це файл у якому малюється наше ігрове поле).

Solver.py – файл з вихідним кодом, в якому, для даного проєкту, за допомогою якого відбувається розв'язування ігрової дошки, тобто ігрового поля.

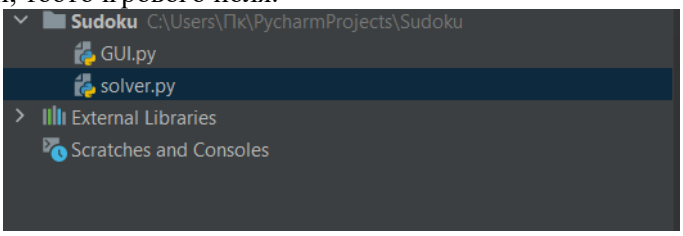


Рисунок 3 - Структура проєкту

Для представлення ігрової дошки використовується двовимірний масив, що складається із 9 стовпців та 9 рядків. Порожні комірки заповнюються 0, а заповнені комірки — відповідними цифрами. Нижче продемонстровано структуру даних для стартової дошки.

```
board = [
    [7,8,0,4,0,0,1,2,0],
    [6,0,0,0,7,5,0,0,9],
    [0,0,0,6,0,1,0,7,8],
    [0,0,7,0,4,0,2,6,0],
    [0,0,1,0,5,0,9,3,0],
    [9,0,4,0,6,0,0,0,5],
    [0,7,0,3,0,0,0,1,2],
    [1,2,0,0,0,7,4,0,0],
    [0,4,9,2,0,6,0,0,7]
]
```

Основними функціями, що описані у файлі Solver.py є valid () та solve().

Функція valid(), синтаксис якої наведено нижче, перевіряє чи є дійсною поточна дошка. На вхід дана функція отримує три параметри:

bo - безпосередньо дошка, яка представляється у вигляді двовимірного масиву цілих чисел;

num - число, що розміщується у задану позицію;

pos - позиція, у яку поміщається задане число num.

```
def valid(bo, num, pos):
    # Перевірка рядка
    for i in range(len(bo[0])):
        if bo[pos[0]][i] == num and pos[1] != i:
            return False

    # Перевірка стовпця
    for i in range(len(bo)):
        if bo[i][pos[1]] == num and pos[0] != i:
            return False

    # Перевірка квадрату 3x3
    box_x = pos[1] // 3
    box_y = pos[0] // 3

    for i in range(box_y*3, box_y*3 + 3):
        for j in range(box_x * 3, box_x*3 + 3):
            if bo[i][j] == num and (i,j) != pos:
                return False

    return True
```

Перевірка правильності дошки здійснюється простим обходом спочатку рядка, потім стовпця, потім квадрату 3x3. У випадку, якщо число num зустрічається у рядку або стовпці, або квадраті, які перевіряються, то функція valid() повертає значення "хибно" (False). Якщо ж число не зустрічається у заданих позиціях, то значення "істинно" (True).

Функція solve(), код якої подано нижче, здійснює розв'язування самого судоку, використовуючи алгоритм бектрекінгу (backtraking).

```
def solve(bo):
    find = find_empty(bo)
    if not find:
        return True
    else:
        row, col = find

    for i in range(1,10):
        if valid(bo, i, (row, col)):
            bo[row][col] = i

            if solve(bo):
                return True

            bo[row][col] = 0

    return False
```

На вхід solve() отримує дошку, де уже розміщені декілька цифр. Надалі функція працює за наступним алгоритмом:

1. Знайти порожню позицію.
2. Спробувати розмістити у ній цифри 1-9.
3. Перевірити, чи може задана цифра може бути розміщена у заданій позиції заданої дошки.
   a. Якщо цифра підходить, рекурсивно заповнити дошку, використовуючи кроки 1-
   b. Якщо цифра не підходить, очистити щойно заповнену комірку (поставте в задану позицію цифру 0), і повернутись до попереднього кроку.

4. Як тільки дошка заповнена визначенням цього алгоритму, дошка розв'язана.

Візуалізація дошки та самого процесу гри описана у файлі GUI.py. У ньому описано клас Grid, який задає структур дошки, та клас Cube, що відповідає за промальовування кожної комірки.

Малювання кожного екземпляру класу cube здійснюється із використанням функції draw(), яка задає товщину ліній для кожної із комірок, розмір та назву шрифту.

Функція click() описує реакцію натискання вказівника миші на поле та дозволяє поставити цифру self у позицію pos.

```
def click(self, pos):
    """
    :param: pos
    :return: (row, col)
    """
    if pos[0] < self.width and pos[1] < self.height:
        gap = self.width / 9
        x = pos[0] // gap
        y = pos[1] // gap
        return (int(y),int(x))
    else:
        return None
```

Врезультаті отримується поле для гри в "Судоку", що складається із 81 клітинки та містить заповнені та порожні поля.

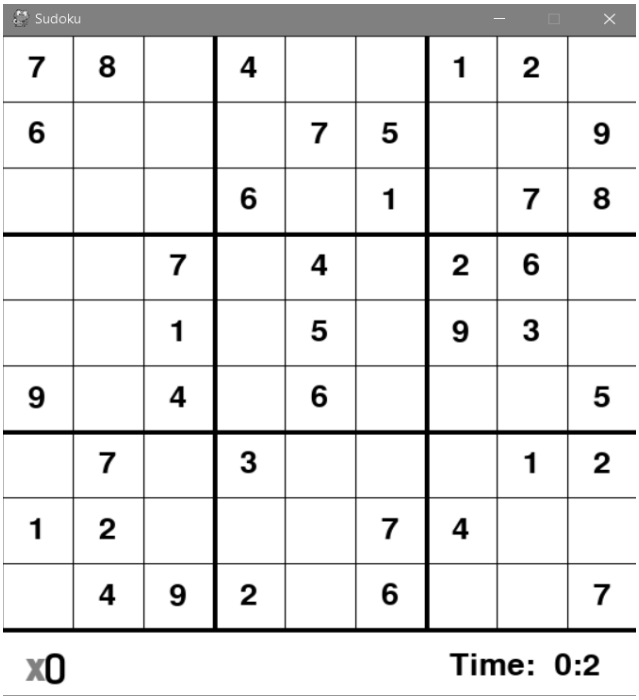
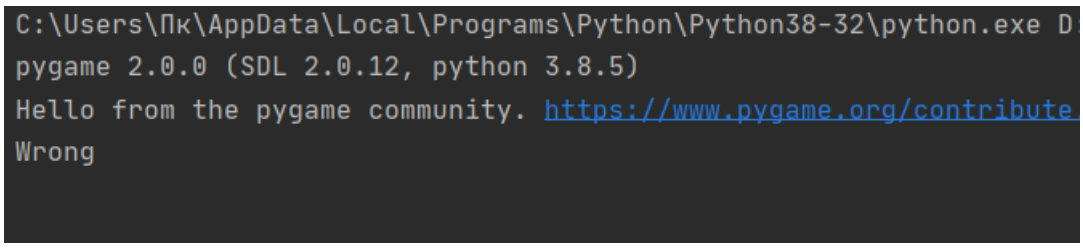


Рисунок 4 – Приклад створеного «Судоку»

У випадку, якщо судоку розв'язане правильно, користувачеві у консоль виводиться відповідне повідомлення.

```
Success
Success
Game over
```

У випадку програшу у консоль також виводиться відповідне повідомлення.



Висновки з дослідження і перспективи подальших розробок. У роботі досліджено розв'язування гри "Судоку", досліджено процес розробки ігрової програми «Судоку» із використанням засобів мови програмування Python. Досліджено середовище розробки програмного забезпечення PyCharm та можливості мови програмування Python для розробки програмного забезпечення. Перспективи для подальшого розвитку програмного продукту полягають у створенні локальної таблиці рекордів для користувачів, у розробці рівнів складності, забезпечення можливості паузи та збереження проміжного варіанту гри для подальшого розв'язування.

Список бібліографічного опису:

1. Bailey R.A., Cameron P.J., Connelly R. Sudoku. Gerechte Designs, Resolutions, Affine Space, Spreads, Reguli, and Hamming Codes // American Mathematical Monthly, Vol. 115, N 5, 2008, pp. 383–404.
2. История судоку [Електронний ресурс]. – 2017. – Режим доступу до ресурсу: https://historygames.ru/golovolomki/istoriya-sudoku.html.
3. Кальчева А. Судоку: японские головоломки [Електронний ресурс] / Анастасия Кальчева // Загадочная Япония. – 2007. – Режим доступу до ресурсу: http://leit.ru/modules.php?name=Pages&pa=showpage&pid=1037&page=1.
4. Муратов С. В. Решаем Судоку (Sudoku) [Електронний ресурс] / Сергей Васильевич Муратов. – 2006. – Режим доступу до ресурсу: http://zhurnal.lib.ru/m/muratow_s_w/sudoku.shtml.
5. Программируем на Python / Майкл Доусон. – Питер, 2019. – 416 с.
6. Простой Python. Современный стиль программирования. / Билл Любанович. – Питер, 2019. – 480 с.
7. https://www.jetbrains.com/ru-ru/pycharm/

References

1. Bailey R.A., Cameron P.J., Connelly R. Sudoku. Gerechte Designs, Resolutions, Affine Space, Spreads, Reguli, and Hamming Codes // American Mathematical Monthly, Vol. 115, No. 5, 2008, pp. 383–404.
2. History of Sudoku [Electronic resource]. - 2017. - Resource access mode: https://historygames.ru/golovolomki/istoriya-sudoku.html.
3. Kalcheva A. Sudoku: Japanese puzzles [Electronic resource] / Anastasia Kalcheva // Mysterious Japan. - 2007. - Resource access mode: http://leit.ru/modules.php?name=Pages&pa=showpage&pid=1037&page=1.
4. Muratov S. V. Solve Sudoku (Sudoku) [Electronic resource] / Sergei Vasilyevich Muratov. - 2006. - Resource access mode: http://zhurnal.lib.ru/m/muratow_s_w/sudoku.shtml.
5. Program in Python / Michael Dawson. - Peter, 2019. - 416 p.
6. Simple Python. Modern programming style. / Bill Lubanovich. - Peter, 2019. - 480 p.
7. https://www.jetbrains.com/ru-ru/pycharm/