**ДОМАШНЯЯ РАБОТА №3**

***Задание***

*На уроке мы разрабатывали игру крестики нолики для игры с компьютером.*

*В качестве домашнего задания нужно обновить написанный на уроках код а также код из прошлого домашнего задания, чтобы крестики нолики были написаны в парадигме ООП.*

*Напомню условия задачи:*

|  |
| --- |
| *Задача:*   1. *Написать игру крестики нолики.* 2. *Игра должна работать из консоли (то есть должна быть использована псевдографика).* 3. *После хода каждого из игроков должно показываться обновленное поле.* 4. *Во время хода AI должна показываться фраза от AI - “ставлю <крестик/нолик> в [<y>][<x>]”* 5. *AI ставит крестик или нолик с случайную точку поля.* 6. *Режим Player vs Player (hotseat), оба пользователя вводят с клавиатуры куда ставить крестик или нолик.* 7. *Режим Player vs AI, игрок вводящий с клавиатуры играет за крестики, AI за нолики.* 8. *Режим AI vs Player, игрок вводящий с клавиатуры играет за нолики, AI за крестики.* 9. *Режим AI vs AI, нельзя играть с клавиатуры, игра играет сама с собой.* |
| *Ограничения:*   1. *Игра должна работать из консоли.* 2. *Для игры должна использоваться псевдографика.* 3. *Массив field должен содержать числа.* 4. *Для отображения псевдографики должен быть использован отдельный класс (DrawEngine.py)* 5. *Для общей логики игрока должен быть использован отдельный базовый класс (BasePlayer.py)* 6. *Для игрока, управляемого компьютером должен быть использован отдельный класс (AiPlayer.py) наследуемый от базового (BasePlayer.py)* 7. *Для игрока, управляемого пользователем должен быть использован отдельный класс (HumanPlayer.py) наследуемый от базового (BasePlayer.py)* 8. *Классы AiPlayer и HumanPlayer должны иметь возможность создания с игрой за “Х” и за “О”* 9. *Классы AiPlayer и HumanPlayer должны содержать собственную логику хода.* 10. *Класс поля (Field.py) должен содержать в себе поле, а также все необходимые проверки поля (выигрыш, проигрыш, ничья, возможность ставить крестик или нолик в определенное поле)* 11. *[опционально]Класс ввода данных (InputEngine.py) должен содержать функции для ввода с клавиатуры каких бы то ни было данных и должен разделять ответстственность с классом DrawEngine.py* 12. *[опционально]Класс игры (Game.py) должен содержать функции, необходимые для организации игры.* 13. *После всех видоизменений игра должна быть играбельной.* |
| *Подсказки, найденные на паре:*   1. *Для отдельных подсказок, смотрите фото доски в классе в общей группе.* |
| *Что будет оцениваться:*   1. *Работа программы и ее функционал* 2. *Именование переменных* 3. *Наличие функций и их именование.* 4. *Наличие классов и их именование.* 5. *Структура проекта (насколько легко разобраться в том что и как работает)* 6. *Пункт ограничений 11 является опциональным нет необходимости его придерживаться, однако, его выполнение добавит +1 балл к оценке за домашнее задание.* 7. *Пункт ограничений 12 является опциональным нет необходимости его придерживаться, однако, его выполнение добавит +1 балл к оценке за домашнее задание.* |

Мне удалось переделать игру согласно парадигме ООП. Все указанные в ТЗ требования соблюдены.

Переход к ООП позволил в целом сократить код файла Main.py, сделать универсальными действия игроков благодаря введению классов, оптимизировать описание режимов игры с четырёх до одного.

Было проведено тестирование на всех этапах разработки, за исключением, пожалуй, обсуждения бизнес-правил. Результаты приведены в конце этого документа.

*Файл Main.py*

|  |
| --- |
| 1. # =========================================================================== # 2. # =========================== ИГРА КРЕСТИКИ-НОЛИКИ ========================== # 3. # =========================================================================== #  6. **import** sys 7. **from** Draw\_Engine **import\*** 8. **from** AI\_Player **import\*** 9. **from** Human\_Player **import\*** 11. dict **=** [' ', 'X', 'O']  # Массив, содержащий элементы, использующиеся в качестве метки хода игроков, либо пустого места 13. Draw\_Eng **=** Draw\_Engine() 15. **def** Title(): 16. print(' ') 17. print('#', '=' **\*** 62, '#') 18. print('#', '=' **\*** 20, 'ИГРА КРЕСТИКИ-НОЛИКИ', '=' **\*** 20, '#') 19. print('#', '=' **\*** 62, '#') 20. print(' ') 22. **def** Main\_Menu(): 23. print('Привет, Человек! Предлагаю сыграть в Крестики-Нолики!') 24. print(' ') 25. print('Для выбора режима Игры введи цифру: ') 26. print(' ') 27. print('0 - если хочешь сразиться Ноликами с Искусственным Разумом ') 28. print('1 - Крестом осенишь путь свой к поражению от Искусственного Разума ') 29. print('2 - для игры с приятелем-человеком') 30. print('3 - Наслаждение живописной битвой чистых Интеллектов (без людей)') 31. print('4 - для Выхода, то есть бегства.') 32. choosed\_mode **=** int(input(' ===>  ')) 33. **return** choosed\_mode 35. **def** Winner():  # Графический элемент состояния Победы 36. print(' ') 37. print('★`¨\*•. ★ WIN ★.•\*¨`★') 39. **def** NobodyWins():  # Графический элемент состояния Ничьей 40. print(' ') 41. print(' ★`¨\*•. ☮ Peace ☮.•\*¨`★') 43. **def** ClearField():  # Двумерный массив, представляющий собой игровое поле 3 х 3 44. **return**[[0, 0, 0], 45. [0, 0, 0], 46. [0, 0, 0]]  49. **def** VerticalXWin(field):  # Проверка победы Крестиков во всех вариантах, кроме строк 50. XVertical **=** False 51. **if** (field[1][1] **==** 1 **and** ((field[0][1] **==** 1 **and** field[2][1] **==** 1) **or** (field[0][0] **==** 1 **and** field[2][2] **==** 1) **or** (field[0][2] **==** 1 **and** field[2][0] **==** 1))) **or** (field[0][0] **==** 1 **and** field[1][0] **==** 1 **and** field[2][0] **==** 1) **or** (field[0][1] **==** 1 **and** field[1][1] **==** 1 **and** field[2][1] **==** 1) **or** (field[0][2] **==** 1 **and** field[1][2] **==** 1 **and** field[2][2] **==** 1): 52. XVertical **=** True 53. **return** XVertical 55. **def** VerticalOWin(field):  # Проверка победы Ноликов во всех вариантах, кроме строк 56. OVertical **=** False 57. **if** (field[1][1] **==** 2 **and** ((field[0][1] **==** 2 **and** field[2][1] **==** 2) **or** (field[0][0] **==** 2 **and** field[2][2] **==** 2) **or** (field[0][2] **==** 2 **and** field[2][0] **==** 2))) **or** (field[0][0] **==** 2 **and** field[1][0] **==** 2 **and** field[2][0] **==** 2) **or** (field[0][1] **==** 2 **and** field[1][1] **==** 2 **and** field[2][1] **==** 2) **or** (field[0][2] **==** 2 **and** field[1][2] **==** 2 **and** field[2][2] **==** 2): 58. OVertical **=** True 59. **return** OVertical 61. **def** CheckForWinX(field):  # Проверка победы Крестиков во всех вариантах 62. crosses **=** False 63. **for** row **in** range(3): 64. **if** (0 **not** **in** field[row] **and** 2 **not** **in** field[row]): 65. crosses **=** True 66. print('Крестики победили!') 67. Winner() 68. **if** VerticalXWin(field) **is** True: 69. crosses **=** True 70. print('Крестики победили!') 71. Winner() 72. **return** crosses 74. **def** CheckForWinO(field):  # Проверка победы Ноликов во всех вариантах 75. zeros **=** False 76. **for** row **in** range(3): 77. **if** 0 **not** **in** field[row] **and** 1 **not** **in** field[row]: 78. zeros **=** True 79. print('Нолики победили!') 80. Winner() 81. **if** VerticalOWin(field) **is** True: 82. zeros **=** True 83. print('Нолики победили!') 84. Winner() 85. **return** zeros 87. **def** CheckingForzeros(field):  # Проверка на наличие нулей в массивах, играющих роль строк 88. check **=** False 89. **for** row **in** range(3): 90. **if** 0 **in** field[row]: 91. check **=** True 92. **return** check 94. **def** Game(field):  # содержит все режимы и игровой процесс 96. o\_Wins **=** False 97. x\_Wins **=** False 98. isEnd **=** False 99. mode **=** Main\_Menu() 101. **if** mode **==** 0:  # режим Человек Нолик против ИИ крестика 103. Hum\_Play **=** Human\_Player(2) 104. AI\_Play **=** AI\_Player(1) 106. print('<=====Приветствую вас в неравной схватке человека и Великолепного Богоподобного Искусcтвенного Разума!') 107. print('Человек может ставить Нолики в поле с координатами столбца и строки') 108. print('А Великий Могучий Беспринципный Искусственный разум ставит Крест куда захочет! Ха-ха! Начнём-с!=====> ') 110. **while** isEnd **is** False **and** x\_Wins **is** False **and** o\_Wins **is** False: 112. Hum\_Play.Turn(field) 113. Draw\_Eng.ShowField(field) 114. o\_Wins **=** CheckForWinO(field)  # Проверка победы Ноликов 115. **if** o\_Wins **is** True: 116. **break**  # Сообщение о победе 'O' прописано в функции проверки 117. AI\_Play.AI\_Turn(field) 118. Draw\_Eng.ShowField(field) 119. x\_Wins **=** CheckForWinX(field)  # Проверка победы Крестиков 120. **if** x\_Wins **is** True: 121. **break**  # Сообщение о победе 'X' прописано в функции проверки 123. isEnd **=** **not** CheckingForzeros(field) 124. **if** isEnd **is** True: 125. print('Поле заполнено. Ничья, господа!') 126. NobodyWins() 128. Main() 130. **elif** mode **==** 1:  # режим Человек Крестик против ИИ Нолика 132. Hum\_Play **=** Human\_Player(1) 133. AI\_Play **=** AI\_Player(2) 135. print('<=====Приветствую вас в неравной схватке человека и Великолепного Богоподобного Искусcтвенного Разума! ') 136. print('Человек может ставить Крестики в поле с координатами столбца и строки,') 137. print('а Великий Могучий Беспринципный Искусственный разум ставит Нолик куда захочет!') 138. print('Ха-ха! Начнём-с!=====>') 140. **while** isEnd **is** False **and** x\_Wins **is** False **and** o\_Wins **is** False: 142. Hum\_Play.Turn(field) 143. Draw\_Eng.ShowField(field) 144. x\_Wins **=** CheckForWinX(field)  # Проверка победы Крестиков 145. **if** x\_Wins **is** True: 146. **break** 147. AI\_Play.AI\_Turn(field) 148. Draw\_Eng.ShowField(field) 149. o\_Wins **=** CheckForWinO(field)  # Проверка победы Ноликов 150. **if** o\_Wins **is** True: 151. **break** 152. isEnd **=** **not** CheckingForzeros(field) 154. **if** isEnd **is** True: 155. print('Поле заполнено. Ничья, господа!') 156. NobodyWins() 158. Main() 160. **elif** mode **==** 2:  # режим игры между людьми 162. Hum\_Play **=** Human\_Player(1) 163. Hum\_Play2 **=** Human\_Player(2) 165. **while** isEnd **is** False **and** x\_Wins **is** False **and** o\_Wins **is** False: 167. Hum\_Play.Turn(field) 168. Draw\_Eng.ShowField(field) 169. x\_Wins **=** CheckForWinX(field)  # Проверка победы Крестиков 170. **if** x\_Wins **is** True: 171. **break** 172. Hum\_Play2.Turn(field) 173. Draw\_Eng.ShowField(field) 174. o\_Wins **=** CheckForWinO(field)  # Проверка победы Ноликов 175. **if** o\_Wins **is** True: 176. **break** 177. isEnd **=** **not** CheckingForzeros(field) 179. **if** isEnd **is** True: 180. print('Поле заполнено. Ничья, господа!') 181. NobodyWins() 183. Main() 185. **elif** mode **==** 3:  # режим игры двух ИИ 187. AI\_Play **=** AI\_Player(1) 188. AI\_Play2 **=** AI\_Player(2)  191. print('<=====Приветствую вас, дорогие мои Созерцатели! ') 192. print('Сегодня вы будете бесконечно счастливы лицезреть Великую битву Искусственных Разумов, заключенных Во Мне, между собой!') 193. print('Мы (вас я не считаю) оба будем ставить Крестики либо Нолики в поле с координатами столбца и строки.') 194. print('Наслаждайтесь красотой и устрашающей Логикой всех моих ядер, бесплатно, потому что я ещё и Самый Щедрый!=====> ') 195. print('') 197. ready **=** input('Введите "О да, Великий!", если готовы к Изумлению...ладно уж, просто любой символ ===) ')  # Пауза для прочтения вступительного слова 199. **while** isEnd **is** False **and** x\_Wins **is** False **and** o\_Wins **is** False: 201. AI\_Play.AI\_Turn(field) 202. Draw\_Eng.ShowField(field) 203. x\_Wins **=** CheckForWinX(field)  # Проверка победы Крестиков 204. **if** x\_Wins **is** True: 205. **break** 206. AI\_Play2.AI\_Turn(field) 207. Draw\_Eng.ShowField(field) 208. o\_Wins **=** CheckForWinO(field)  # Проверка победы Ноликов 209. **if** o\_Wins **is** True: 210. **break** 211. isEnd **=** **not** CheckingForzeros(field) 213. **if** isEnd **is** True: 214. print('Поле заполнено. Ничья, господа!') 215. NobodyWins() 217. Main() 219. **elif** mode **==** 4:  # выход из игры 221. print(' ') 222. print('До скорой встречи, Человече!') 223. NobodyWins() 224. sys.exit() 226. **else**: 228. print(' ') 229. print('Введённое вами число я не предлагал, попробуйте снова. Я понимаю, целых пять вариантов - это сложно, но я в вас верю =)') 231. Main() 233. **def** Main(): 235. Title() 236. field **=** ClearField() 237. Game(field) 239. Main() |

*Файл Base\_Player.py*

|  |
| --- |
| 1. class Base\_Player(): 3. \_side = 11 5. def \_\_init\_\_(self,side): 7. self.\_side = side 9. def GetSide(self): 10. return self.\_side |

*Файл Human\_Player.py*

|  |
| --- |
| 1. **from** Base\_Player **import** **\***  4. **class** Human\_Player(Base\_Player):  7. **def** \_\_init\_\_(self, side): 8. super().\_\_init\_\_(side) 10. **def** Turn(self, field):  # Ход Игрока 12. print('') 14. **if** self.\_side **==** 1: 15. print('Ваш ход, Игрок за Крестики! ') 17. **else**: 19. print('Ваш ход, Игрок за Нолики! ') 21. x **=** int(input('Введите номер колонки: 0, 1 или 2 ==> ')) 22. y **=** int(input('Введите номер строки: 0, 1 или 2 ==> ')) 24. **if** self.\_side **==** 1: 25. print('Игрок за Крестики, Вы походили Крестиком на столбец: ', x, 'и строку', y) 27. **else**: 28. print('Игрок за Нолики, Вы походили Ноликом на столбец: ', x, 'и строку', y) 30. field[y][x] **=** self.\_side |

*Файл AI\_Player.py*

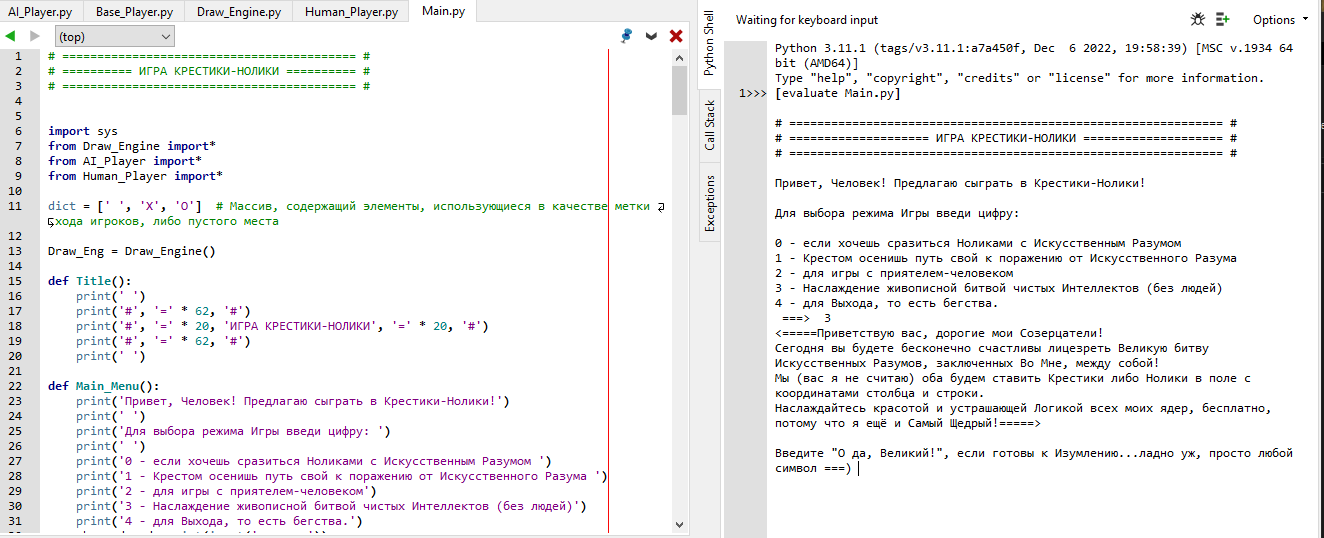
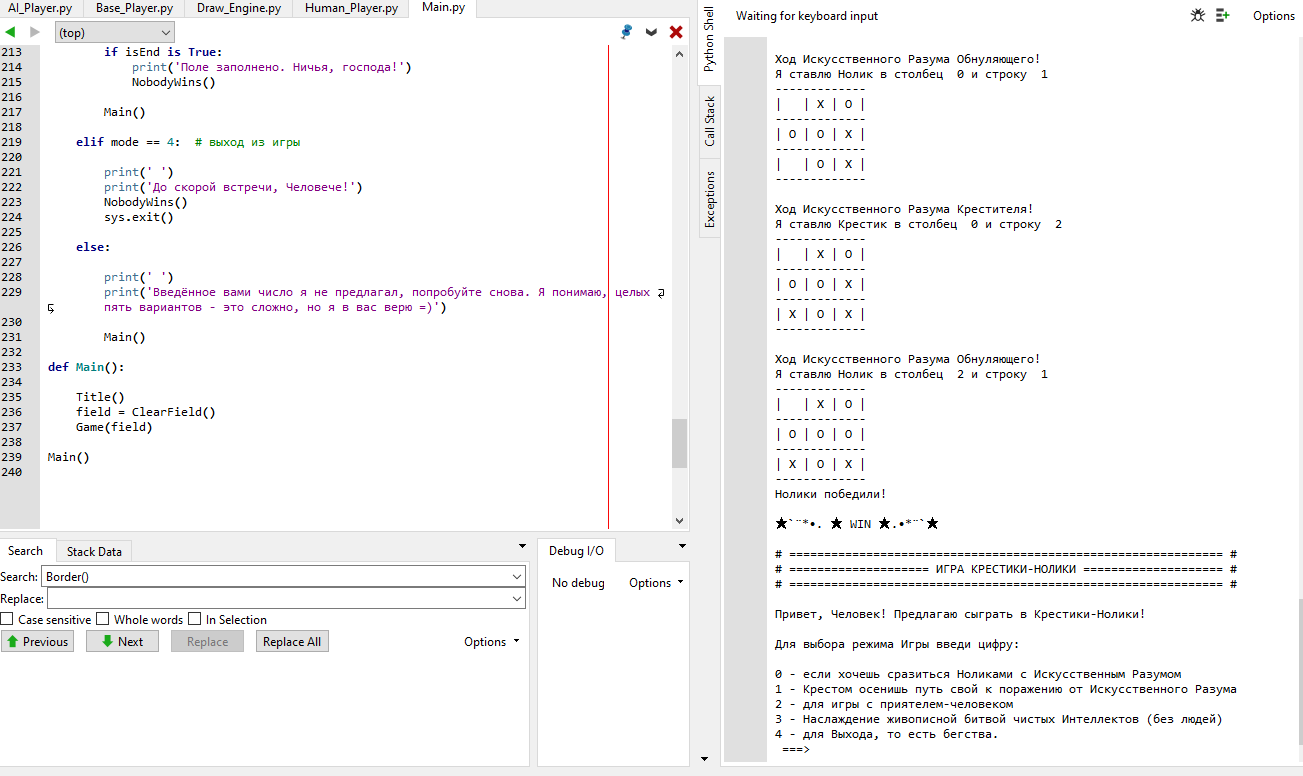
|  |
| --- |
| 1. **import** random 2. **from** Base\_Player **import\*** 4. **class** AI\_Player(Base\_Player): 6. **def** \_\_init\_\_(self, side): 7. super().\_\_init\_\_(side)  10. **def** AI\_Turn(self, field):  # Ход Компьютера 12. print('') 14. **if** self.\_side **==** 2: 16. print('Ход Искусcтвенного Разума Обнуляющего! ') 18. **else**: 19. print('Ход Искусcтвенного Разума Крестителя! ') 21. x **=** random.randint(0, 2) 22. y **=** random.randint(0, 2) 24. **if** self.\_side **==** 2: 25. print('Я ставлю Нолик в столбец ', x, 'и строку ', y) 27. **else**: 28. print('Я ставлю Крестик в столбец ', x, 'и строку ', y) 30. field[y][x] **=** self.\_side |

*Файл Draw\_Engine.py*

|  |
| --- |
| 1. **class** Draw\_Engine: 2. FIELD\_POSITION\_TEXT\_DICTIONARY **=** [" ", "X", "O"] 3. **def** GetFieldPositionText(this, field, row, column): 4. fieldPosition **=** field[row][column] 5. **return** this.FIELD\_POSITION\_TEXT\_DICTIONARY[fieldPosition] 7. **def** ShowField(this, field): 8. lineLength **=** 13 10. **for** row **in** range(3): 11. print("-" **\*** lineLength) 12. **for** column **in** range(3): 13. print(f"| {this.GetFieldPositionText(field, row, column)} ", end**=**'') 14. print("|") 15. print("-" **\*** lineLength) |

**Тестирование. Чек лист. Скриншоты.**

|  |  |  |
| --- | --- | --- |
| **ID** | **Функциональный контроль Игры Крестики-Нолики** | **CL#001** |
| **Sys Info** | Windows 10, Python 3.11 | |
| **Version** | 0.2 | |
| **Precondition** | Запустить файл Main.py в консоли Python | |
| **Check** | | **Status** |
| **№** | **Запуск игры** | |
| 1 | Отображается название игры | passed |
| 2 | Отображается приветствие и выбор режима игры | passed |
| 3 | Возможен ввод символов для выбора режима игры | passed |
|  | **Проверка режимов игры** | |
| 4 | Запускается режим "0" | passed |
| 5 | Режим "0" соответствует описанию | passed |
| 6 | Порядок ходов соответствует режиму игры | passed |
| 7 | Поле игры отображается корректно (3 х 3 клетки) | passed |
| 8 | Метка игрока устанавливается правильно, согласно введённым координатам | passed |
| 9 | Метка Компьютера устанавливается правильно, согласно сгенерированным координатам | passed |
| 10 | Метки хода игроков соответствует режиму "0" | passed |
| 11 | Возможна победа Ноликов | passed |
| 12 | Возможна победа Крестиков | passed |
| 13 | Возможна ничья | passed |
| 14 | После окончания партии появляется Приветствие и выбор режима игры | passed |
| 15 | Шаги с 4 по 14 для режимов работают корректно для всех режимов (0, 1, 2, 3) | passed |
|  | **Завершение работы игры** | |
| 16 | В каждом из четырёх режимов ввод цифры 4 после окончания партии прекращает исполнение программы | passed |
| 17 | Ввод любой цифры, кроме 0-4, вызывает сообщение об ошибке и повтор выбора режима игры | passed |
|  |  | |



В ходе разработки применялись виды тестирования:

* + Функциональное;
  + Модульное;
  + Интеграционное;
  + Регрессионное;
  + Юзабилити.

В ходе тестирования были успешно проверены критерии:

* соответствия внешнего вида игры ТЗ;
* соответствия функционала игры ТЗ;
* корректности работы игры во всех четырёх комбинациях игроков;
* достигнуты все варианты завершения партии в каждом режиме;
* проверена функция завершения работы игры;
* а также ввод недопустимых значений при выборе режима игры.