Қазақстан Республикасы Оқу-ағарту министрлігі

«Білім» кәсіби гуманитарлық-техникалық колледжі



**Зертханалық жұмыс №1**

**Тақырыбы: Шарттарды программалау және PyGame кітапханасын қосу**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Жұмыстың орындау сапасы | Баға диапазоны | Орындаған % |
| 1 | Орындалған жоқ, сабақта себепсіз болмады. | 0 % |  |
| 2 | Жұмыстың орындалуы және студенттің белсенділігі | 0-50% |  |
| 3 | Жұмысты рәсімдеу | 0-20% |  |
| 4 | Анықтамалар мен техникалық әдістемелерді, пәннің оқу-әдістемелік кешенін, лекция конспектілерін қолдана білу. | 0-5% |  |
| 5 | Техникалық құралдарды пайдалана білу | 0-5% |  |
| 6 | Жұмысты қорғау | 0-20% |  |
|  | Қорытынды | 0-100% |  |

Оқытушы: Нургисаева У.М.

Студент:Саду Айжамал

Мамандығы:Бағдарламалық қаматамасыз ету

Тараз 2025

**Зертханалық жұмыс №1**

**Тақырыбы**: Шарттарды программалау және PyGame кітапханасын қосу

**Мақсаты:** Студенттерге шартты операторлар мен логикалық операторларды қолдануды үйрету. **PyGame** кітапханасы арқылы ойындар жасау негіздерін меңгерту.Шартты операторлар мен PyGame-ды біріктіру арқылы қарапайым ойын логикасын құруды үйрету.

**Міндеттері:**

* PyGame кітапханасын орнату және конфигурациялау:
* Қарапайым ойын логикасын құру:
* Қарапайым анимацияларды жасау:
* Қателермен жұмыс істеу:

**Қажетті құралдар:**

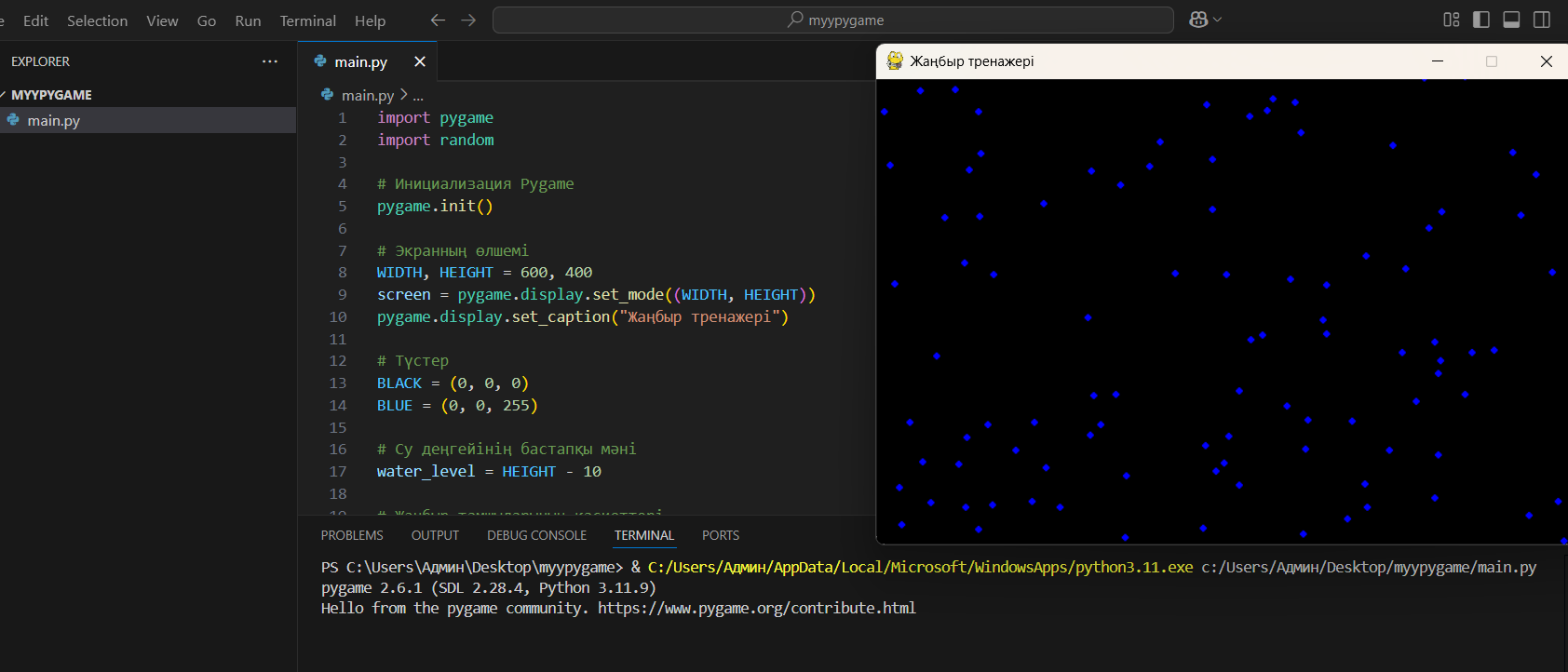
1. Бағдарламалау ортасы: VS Code
2. Дербес компьютер немесе ноутбук

**1-Тапсырма-Жаңбыр**

Pygame көмегімен жаңбыр тренажерін жазыңыз: әр жүз тамшының құлауы су деңгейінің 1 пиксельге көтерілуіне әкеледі.

**Задание 1 – Дождь**

Используя Pygame, напишите симулятор дождя: падение каждой сотни капель приводит к подъему уровня воды на 1 пиксель.



Әр код жолының түсіндірмесі:

1. \*import pygame\*: Pygame кітапханасын қосу. Бұл кітапхана ойындар мен графикалық интерфейстер жасау үшін қолданылады.

2. \*import random\*: Кездейсоқ сандарды генерациялау үшін random кітапханасын қосу.

3. \*pygame.init()\*: Pygame кітапханасын инициализациялау, бұл барлық компоненттерді жұмысқа дайындайды.

4. \*WIDTH, HEIGHT = 600, 400\*: Экранның ені мен биіктігін анықтайды. Бұл параметрлер экранның өлшемін белгілейді.

5. \*screen = pygame.display.set\_mode((WIDTH, HEIGHT))\*: Экран терезесін жасау және оның өлшемін орнату.

6. \*pygame.display.set\_caption("Жаңбыр тренажері")\*: Терезенің тақырыбын орнату.

7. \*BLACK = (0, 0, 0)\* және \*BLUE = (0, 0, 255)\*: Түстерді анықтау. BLACK - қара түс, ал BLUE - көк түс.

8. \*water\_level = HEIGHT - 10\*: Су деңгейінің бастапқы орнын орнату, экранның төменгі жағынан 10 пиксель жоғары.

9. \*raindrops = []\*: Жаңбыр тамшыларын сақтайтын тізімді анықтау.

10. \*RAINDROP\_COUNT = 100\*: Жаңбыр тамшыларының жалпы санын 100 деп орнату.

11. \*class Raindrop:\*: Жаңбыр тамшысының сыныбын анықтау. Бұл сынып әрбір тамшының мінез-құлқын анықтайды.

12. \*def \_\_init\_\_(self):\*: Жаңбыр тамшысының бастапқы орнын және жылдамдығын анықтау.

- \*self.x = random.randint(0, WIDTH)\*: Тамшының көлденең координатасын кездейсоқ таңдау.

- \*self.y = random.randint(-20, HEIGHT)\*: Тамшының вертикальды координатасын кездейсоқ таңдау.

- \*self.speed = random.randint(2, 5)\*: Тамшының жылдамдығын кездейсоқ таңдау.

13. \*def update(self):\*: Жаңбыр тамшысының жаңартылуын анықтайтын функция.

- \*self.y += self.speed\*: Тамшының төмен түсуін есептеу.

- \*if self.y > HEIGHT:\*: Егер тамшы экранның төменгі шекарасына жетсе, оның орнын қайтадан жоғарғы жағынан бастау.

- \*return True\*: Су деңгейін көтеретінін көрсету.

- \*return False\*: Егер тамшы экранның төменгі жағынан өтпесе, онда ештеңе өзгермейді.

14. \*running = True: Негізгі циклдың жалғасуын басқаратын айнымалы. \*\*False\* болса, цикл тоқтайды.

15. \*while running:\*: Негізгі цикл басталады, ол бағдарлама жұмыс істей бергенде орындалады.

16. \*for event in pygame.event.get():\*: Экрандағы барлық оқиғаларды тексеру.

- \*if event.type == pygame.QUIT:\*: Егер терезе жабылса, бағдарлама тоқтайды.

17. \*for raindrop in raindrops:\*: Барлық жаңбыр тамшыларын тексеру.

- \*if raindrop.update():\*: Әрбір тамшының орнын жаңарту және егер тамшы экранның төменгі жағына жетсе, су деңгейін көтеру.

- \*water\_level += 1\*: Су деңгейін 1 пиксельге көтеру.

18. \*screen.fill(BLACK)\*: Экранды қара түспен толтыру, бұл экранды тазалауды білдіреді.

19. \*pygame.draw.rect(screen, BLUE, (0, water\_level, WIDTH, HEIGHT - water\_level))\*: Су деңгейін көк түспен көрсету үшін тікбұрыш сызу.

20. \*for raindrop in raindrops:\*: Барлық жаңбыр тамшыларын экранға салу.

- \*pygame.draw.circle(screen, BLUE, (raindrop.x, raindrop.y), 3)\*: Әрбір тамшыны көк түспен шеңбер ретінде көрсету.

21. \*pygame.display.flip()\*: Экранды жаңарту (барлық өзгерістерді көрсетеді).

22. \*if len(raindrops) < RAINDROP\_COUNT:\*: Егер жаңбыр тамшыларының саны 100-ден аз болса, жаңа тамшы қосу.

- \*raindrops.append(Raindrop())\*: Жаңа жаңбыр тамшысын тізімге қосу.

23. \*pygame.time.Clock().tick(60)\*: FPS жылдамдығын 60-қа орнату, бұл секундына 60 кадрды қамтамасыз етеді.

24. \*pygame.quit()\*: Pygame кітапханасын жабу, ресурстарды босату.

Бұл код жаңбыр тамшыларының экранында құлап, әрбір құлаудың су деңгейін көтеруін көрсетуге арналған тренажер болып табылады.Әр код жолының түсіндірмесі:

1. \*import pygame\*: Pygame кітапханасын қосу. Бұл кітапхана ойындар мен графикалық интерфейстер жасау үшін қолданылады.

2. \*import random\*: Кездейсоқ сандарды генерациялау үшін random кітапханасын қосу.

3. \*pygame.init()\*: Pygame кітапханасын инициализациялау, бұл барлық компоненттерді жұмысқа дайындайды.

4. \*WIDTH, HEIGHT = 600, 400\*: Экранның ені мен биіктігін анықтайды. Бұл параметрлер экранның өлшемін белгілейді.

5. \*screen = pygame.display.set\_mode((WIDTH, HEIGHT))\*: Экран терезесін жасау және оның өлшемін орнату.

6. \*pygame.display.set\_caption("Жаңбыр тренажері")\*: Терезенің тақырыбын орнату.

7. \*BLACK = (0, 0, 0)\* және \*BLUE = (0, 0, 255)\*: Түстерді анықтау. BLACK - қара түс, ал BLUE - көк түс.

8. \*water\_level = HEIGHT - 10\*: Су деңгейінің бастапқы орнын орнату, экранның төменгі жағынан 10 пиксель жоғары.

9. \*raindrops = []\*: Жаңбыр тамшыларын сақтайтын тізімді анықтау.

10. \*RAINDROP\_COUNT = 100\*: Жаңбыр тамшыларының жалпы санын 100 деп орнату.

11. \*class Raindrop:\*: Жаңбыр тамшысының сыныбын анықтау. Бұл сынып әрбір тамшының мінез-құлқн анықтайды.

12. \*def \_\_init\_\_(self):\*: Жаңбыр тамшысының бастапқы орнын және жылдамдығын анықтау.

- \*self.x = random.randint(0, WIDTH)\*: Тамшының көлденең координатасын кездейсоқ таңдау.

- \*self.y = random.randint(-20, HEIGHT)\*: Тамшының вертикальды координатасын кездейсоқ таңдау.

- \*self.speed = random.randint(2, 5)\*: Тамшының жылдамдығын кездейсоқ таңдау.

13. \*def update(self):\*: Жаңбыр тамшысының жаңартылуын анықтайтын функция.

- \*self.y += self.speed\*: Тамшының төмен түсуін есептеу.

- \*if self.y > HEIGHT:\*: Егер тамшы экранның төменгі шекарасына жетсе, оның орнын қайтадан жоғарғы жағынан бастау.

- \*return True\*: Су деңгейін көтеретінін көрсету. - \*return False\*: Егер тамшы экранның төменгі жағынан өтпесе, онда ештеңе өзгермейді.

14. \*running = True: Негізгі циклдың жалғасуын басқаратын айнымалы. \*\*False\* болса, цикл тоқтайды.

15. \*while running:\*: Негізгі цикл басталады, ол бағдарлама жұмыс істей бергенде орындалады.

16. \*for event in pygame.event.get():\*: Экрандағы барлық оқиғаларды тексеру.

- \*if event.type == pygame.QUIT:\*: Егер терезе жабылса, бағдарлама тоқтайды.

17. \*for raindrop in raindrops:\*: Барлық жаңбыр тамшыларын тексеру.

- \*if raindrop.update():\*: Әрбір тамшының орнын жаңарту және егер тамшы экранның төменгі жағына жетсе, су деңгейін көтеру.

- \*water\_level += 1\*: Су деңгейін 1 пиксельге көтеру.

18. \*screen.fill(BLACK)\*: Экранды қара түспен толтыру, бұл экранды тазалауды білдіреді.

19. \*pygame.draw.rect(screen, BLUE, (0, water\_level, WIDTH, HEIGHT - water\_level))\*: Су деңгейін көк түспен көрсету үшін тікбұрыш сызу.

20. \*for raindrop in raindrops:\*: Барлық жаңбыр тамшыларын экранға салу.

- \*pygame.draw.circle(screen, BLUE, (raindrop.x, raindrop.y), 3)\*: Әрбір тамшыны көк түспен шеңбер ретінде көрсету.

21. \*pygame.display.flip()\*: Экранды жаңарту (барлық өзгерістерді көрсетеді).

22. \*if len(raindrops) < RAINDROP\_COUNT:\*: Егер жаңбыр тамшыларының саны 100-ден аз болса, жаңа тамшы қосу.

- \*raindrops.append(Raindrop())\*: Жаңа жаңбыр тамшысын тізімге қосу.

23. \*pygame.time.Clock().tick(60)\*: FPS жылдамдығын 60-қа орнату, бұл секундына 60 кадрды қамтамасыз етеді.

24. \*pygame.quit()\*: Pygame кітапханасын жабу, ресурстарды босату.

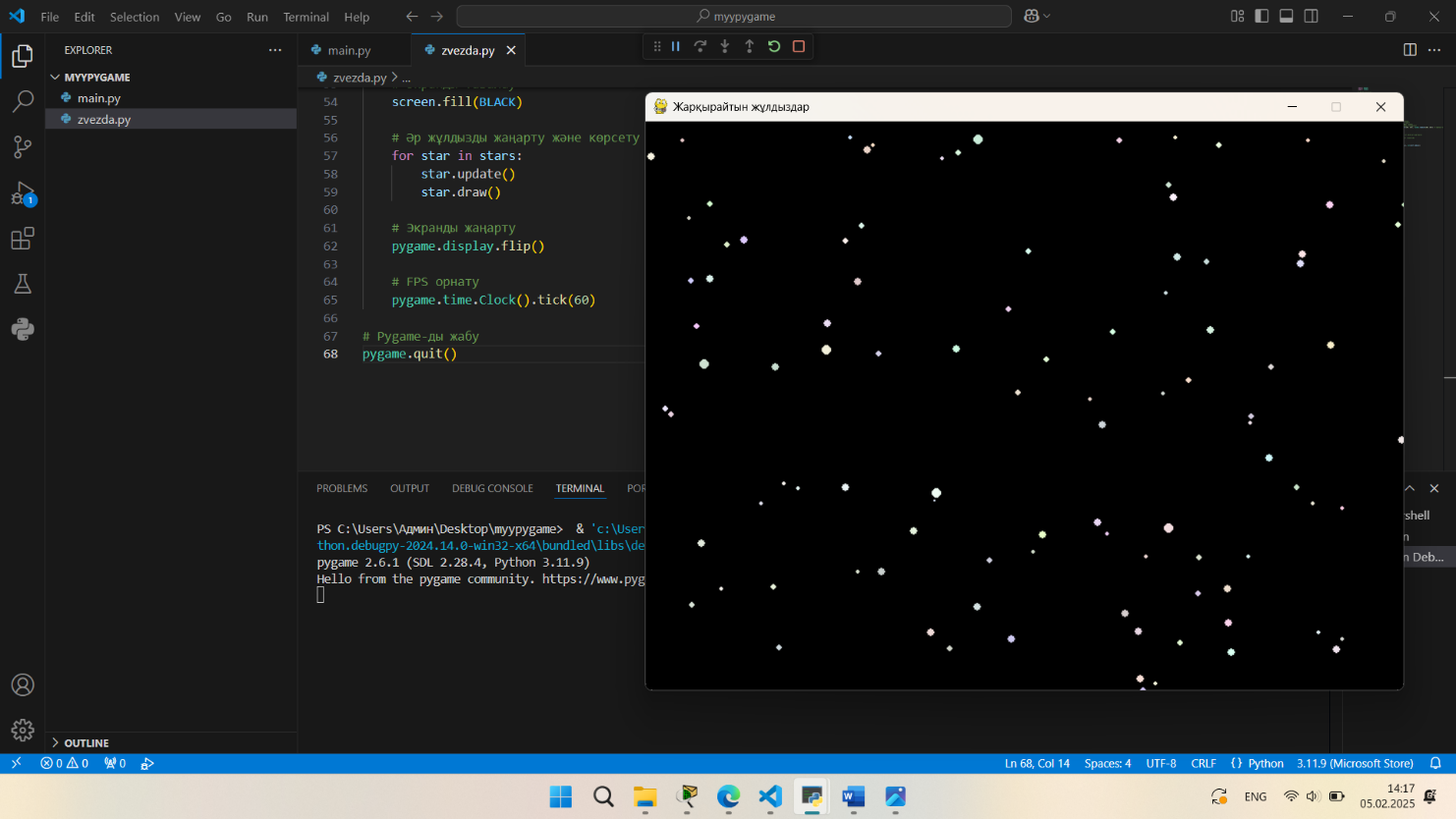
Бұл код жаңбыр тамшыларының экранында құлап, әрбір құлаудың су деңгейін көтеруін көрсетуге арналған тренажер болып табылады.

**2-Тапсырма - Жарқырайтын жұлдыздар**

Pygame көмегімен жұлдызды аспан тренажерін жазыңыз-жұлдыздар болып табылатын шеңберлер жыпылықтауды имитациялау үшін кішірейіп, кеңейеді.

**Задание 2 – Мерцающие звезды**

Используя Pygame, напишите симулятор звездного неба – окружности, представляющие собой звезды, сжимаются и расширяются, имитируя мерцание.



Бұл бағдарлама Pygame кітапханасын пайдаланып, жарқыраған жұлдыздардың анимациясын жасайды. Төменде әрбір кодтың жолдарын түсіндіре отырып, реферат түрінде сипаттаймын.

Бағдарлама бастамасында, алдымен Pygame және кездейсоқ сандарды генерациялау үшін pygame мен random кітапханалары импортталады. Pygame кітапханасын қолдануға дайын болу үшін pygame.init() әдісі арқылы инициализация жасалады.

Кейін экранның өлшемдері анықталады: экранның ені 800 пиксель, ал биіктігі 600 пиксель. Бұл параметрлерді pygame.display.set\_mode() әдісі арқылы орнатып, экранды жасайды. Экран терезесінің атауы "Жарқырайтын жұлдыздар" деп белгіленеді.

Түстерді анықтау үшін қара (BLACK) және ақ (WHITE) түстердің RGB мәндері беріледі. Бұл түстер экранды бояу үшін пайдаланылады.

Жұлдыздар үшін бос тізім (stars) жасалады, ал жұлдыздардың жалпы саны 100 деп орнатылады. Осыдан кейін жұлдыздарды сипаттайтын Star атты класс анықталады. Бұл класс әр жұлдыздың координаттарын, радиусын, жылдамдығын және түсін сипаттайды.

Star класында \_\_init\_\_ әдісі жұлдыздың бастапқы параметрлерін кездейсоқ түрде орнатады. Әр жұлдыздың x және y координаталары экранның шекараларында кездейсоқ таңдалады. Радиус 2 мен 5 арасында, жылдамдық 0.02 мен 0.1 арасында ауытқиды. Түс ретінде кездейсоқ жарық түстері таңдалады.

Жұлдыздардың анимациясын жүзеге асыратын update әдісі бар. Бұл әдіс арқылы жұлдыздың радиусы ұлғайып, кішірейіп отырады. Егер радиус 5-ке жетсе, оның жылдамдығы кері бағытта өзгеріп, кішірейе бастайды. Ал радиус 2-ге жеткенде, оның жылдамдығы қайтадан өсіп, радиус ұлғаяды.

Жұлдыздарды экранға шығару үшін draw әдісі қолданылады. Бұл әдіс жұлдыздың орнын, түсін және радиусын пайдаланып, экранға дөңгелек салу үшін pygame.draw.circle() әдісін қолданады.

Негізгі циклде Star класының 100 объектісі жасалып, олар тізімге қосылады. Бағдарламаның негізгі жұмыс циклі ішінде әрбір жұлдыздың позициясы жаңартылады және экранға салынады. Экранды әрбір кадр сайын жаңарту үшін pygame.display.flip() әдісі пайдаланылады. Сонымен қатар, pygame.time.Clock().tick(60) арқылы әр секундта 60 кадр жаңартылады, бұл анимацияның тегіс әрі тұрақты жүруін қамтамасыз етеді.

Барлық жұлдыздардың анимациясы орындалып болған соң, Pygame кітапханасы арқылы жұмыс аяқталады.

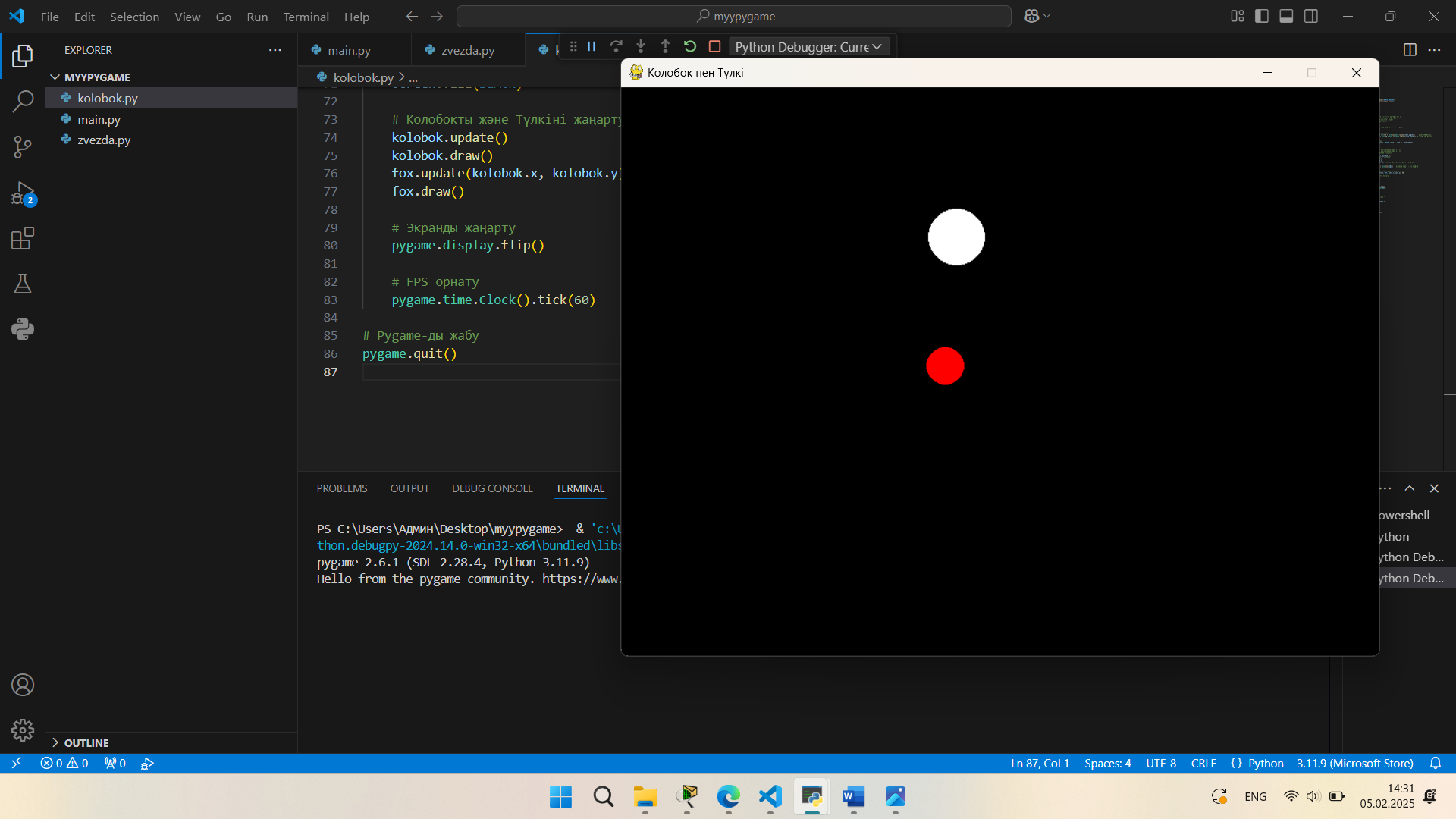
Бұл бағдарлама Pygame арқылы динамикалық және көркем анимацияны жасау негіздерін түсіндіреді. Жұлдыздардың әртүрлі түстер мен өлшемдерде жарқырауы визуалды әсерлерді байытады.

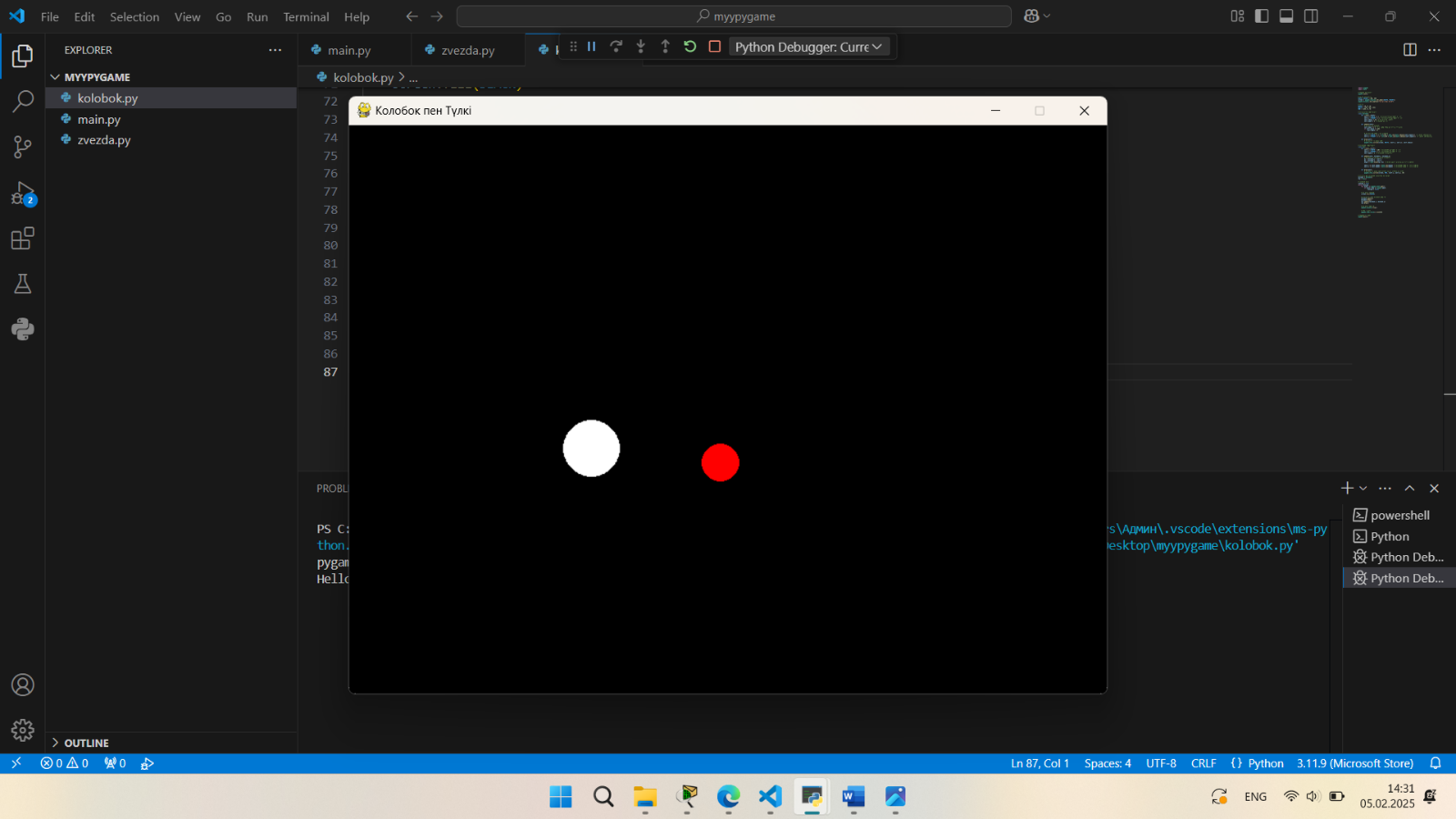
**3-Тапсырма - Колобок**

Pygame көмегімен түлкі (осы кадрлардан тұратын) колобокты қуып жететін анимация жасаңыз. Колобок өз осінің айналасында айналады.

**Задание 3 – Колобок**

Используя Pygame, создайте анимацию, в которой лиса ([состоящая из этих фреймов](https://github.com/natkaida/pygame_projects/tree/main/zadanie_5_kolobok/assets)) преследует [Колобка](https://github.com/natkaida/pygame_projects/blob/main/zadanie_5_kolobok/assets/kolobok.png). Колобок вращается вокруг своей оси.





Әр код жолының түсіндірмесі:

1. \*import pygame\*: Pygame кітапханасын қосу. Бұл кітапхана ойындар мен анимация жасау үшін қолданылады.

2. \*import math\*: Математикалық функцияларды қолдану үшін math кітапханасын қосу. Бұл кітапхана тригонометриялық функциялар сияқты пайдалы құралдар береді.

3. \*pygame.init()\*: Pygame кітапханасын инициализациялау. Барлық Pygame компоненттерін іске қосады.

4. \*WIDTH, HEIGHT = 800, 600\*: Экранның ені мен биіктігін орнату. Экранның өлшемдерін 800x600 пиксель етіп белгілейміз.

5. \*screen = pygame.display.set\_mode((WIDTH, HEIGHT))\*: Экран терезесін жасау. Экранның ені мен биіктігін WIDTH және HEIGHT мәндеріне тең етіп орнатады.

6. \*pygame.display.set\_caption("Түлкі мен Колобок")\*: Экран терезесінің атын орнату. Бұл атау терезенің жоғарғы жағында көрсетіледі.

7. \*BLACK = (0, 0, 0)\*: Қара түс үшін RGB мәнін анықтау. Экранды тазалау үшін қолданамыз.

8. \*ORANGE = (255, 165, 0)\*: Колобоктың түсін анықтау (сары түс).

9. \*BROWN = (139, 69, 19)\*: Түлкінің түсін анықтау (қоңыр түс).

10. \*fox\_x, fox\_y = 300, 300\*: Түлкінің бастапқы координаттары. Бұл түлкінің экранда орналасқан орнын анықтайды.

11. \*ball\_x, ball\_y = WIDTH // 2, HEIGHT // 2\*: Колобоктың бастапқы координаттары. Бұл колобоктың экранның ортасында орналасқанын көрсетеді.

12. \*ball\_radius = 30\*: Колобоктың радиусы. Бұл оның диаметрі жартысынан табылады.

13. \*angle = 0\*: Колобоктың айналу бұрышы. Бұл параметр әр циклда артып, колобоктың айналатын бағытын басқарады.

14. \*fox\_speed = 2\*: Түлкінің жылдамдығы. Бұл түлкінің қозғалыс жылдамдығын анықтайды

15. \*ball\_speed = 0.05\*: Колобоктың айналу жылдамдығы. Бұл параметр колобоктың қаншалықты тез айналатынын анықтайды.

16. \*running = True\*: Бағдарламаның жұмыс істеуін басқару. Егер бұл мән False болса, бағдарлама тоқтайды.

17. \*while running:\*: Негізгі цикл. Бұл цикл барлық анимацияны басқарады және экранды үнемі жаңартып тұрады.

18. \*for event in pygame.event.get():\*: Барлық оқиғаларды тексеру. Бұл ойынның терезесі жабылғанын немесе басқа оқиғаларды өңдеуге мүмкіндік береді.

19. \*if event.type == pygame.QUIT:\*: Егер терезе жабылса, бағдарлама тоқтайды.

20. \*angle += ball\_speed\*: Колобоктың айналу бұрышын арттыру. Бұл әр цикл сайын колобоктың айналуын іске қосады.

21. \*if angle >= 2 \* math.pi:\*: Егер бұрыш 360 градусқа жетсе, яғни бір айналым толық болса, бұрышты қайтадан 0-ге теңестіру.

22. \*ball\_x = WIDTH // 2 + math.cos(angle) \* 100\*: Колобоктың көлденең координатасын есептеу. Колобок экранның ортасынан 100 пиксель қашықтықта айналады, сондықтан оның орнын тригонометриялық функциялармен есептейміз.

23. \*ball\_y = HEIGHT // 2 + math.sin(angle) \* 100\*: Колобоктың вертикаль координатасын есептеу. Колобоктың айналуының нәтижесінде оның y-координаты да өзгереді.

24. \*if fox\_x < ball\_x:\*: Егер түлкінің көлденең координаты колобоктың көлденең координатынан кіші болса, ол оңға қарай қозғалады.

25. \*fox\_x += fox\_speed\*: Түлкінің көлденең координатын арттыру. Бұл түлкіні оңға жылжытады.

26. \*if fox\_x > ball\_x:\*: Егер түлкінің көлденең координаты колобоктың көлденең координатынан үлкен болса, ол солға қарай қозғалады.

27. \*fox\_x -= fox\_speed\*: Түлкінің көлденең координатын кеміту. Бұл түлкіні солға жылжытады.

28. \*if fox\_y < ball\_y:\*: Егер түлкінің вертикаль координаты колобоктың вертикаль координатынан кіші болса, ол төмен қарай қозғалады.

29. \*fox\_y += fox\_speed\*: Түлкінің вертикаль координатасын арттыру. Бұл түлкіні төмен жылжытады.

30. \*if fox\_y > ball\_y:\*: Егер түлкінің вертикаль координаты колобоктың вертикаль координатынан үлкен болса, ол жоғары қарай қозғалады.

31. \*fox\_y -= fox\_speed\*: Түлкінің вертикаль координатасын кеміту. Бұл түлкіні жоғары жылжытады.

32. \*screen.fill(BLACK)\*: Экранды қара түспен толтыру. Бұл экранды әр циклда тазалайды.

33. \*pygame.draw.circle(screen, ORANGE, (int(ball\_x), int(ball\_y)), ball\_radius)\*: Колобокты экранға салу. Бұл оның орнын және радиусын пайдаланып сары дөңгелек ретінде салады.

34. \*pygame.draw.rect(screen, BROWN, (fox\_x - 15, fox\_y - 15, 30, 30))\*: Түлкіні экранға салу. Түлкіні қарапайым төртбұрыш ретінде саламыз.

35. \*pygame.display.flip()\*: Экранды жаңарту. Барлық өзгерістерді көрсетеді

36. \*pygame.time.Clock().tick(60)\*: FPS орнату. Бұл бағдарлама 60 кадр/секунд жылдамдықпен жұмыс істейді.

37. \*pygame.quit()\*: Pygame кітапханасын жабу. Барлық ресурстарды босату.

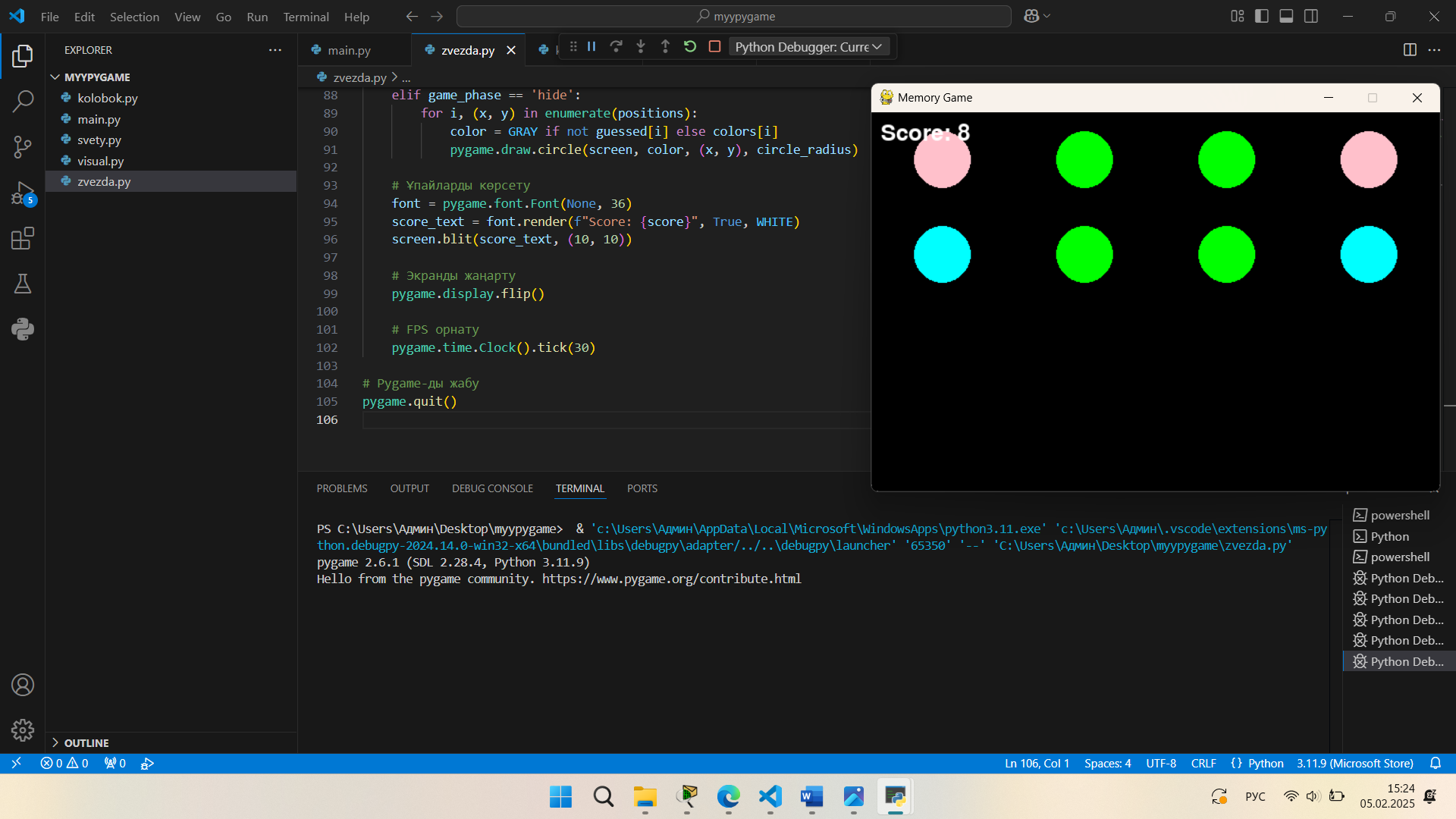
Бұл кодтың нәтижесінде, колобок өз осінің айналасында айналып, түлкі оны қуып жетуге тырысады.

**4-Тапсырма - Визуалды жад**

Pygame мүмкіндіктерін пайдаланып Memory game ойынының жеңіл нұсқасын жазыңыз. Алдымен бағдарлама түрлі-түсті шеңберлерді шығарады (кездейсоқ ретпен) және пайдаланушыға олардың орналасқан жерін бірнеше секунд ішінде есте сақтауға мүмкіндік береді. Содан кейін бағдарлама түрлі-түсті шеңберлерді сұр түспен жабады: пайдаланушы жад бойынша түрлі-түсті жұптарды сәйкестендіруі керек. Әрбір болжамды жұп пайдаланушыға 1 ұпай береді.

**Задание 4 – Визуальная память**

Напишите лайт-версию игры Memory game, используя возможности Pygame. Сначала приложение выводит (в случайном порядке) цветные окружности и дает возможность пользователю запомнить их расположение в течение нескольких секунд. Затем приложение закрывает цветные окружности серыми: пользователь должен по памяти сопоставить цветные пары. Каждая угаданная пара приносит пользователю 1 балл.



1. **import pygame** - Pygame кітапханасын импорттау. Бұл кітапхана ойындар жасау үшін қажет құралдарды қамтамасыз етеді.
2. **import random** - Random кітапханасын импорттау. Бұл кітапхана кездейсоқ сандар мен элементтерді генерациялау үшін пайдаланылады.
3. **import time** - Time кітапханасын импорттау. Бұл кітапхана уақытты басқару үшін қолданылады, мысалы, ұйқы режиміне қою.
4. **pygame.init()** - Pygame-ды бастау. Барлық Pygame модульдері мен мүмкіндіктерін іске қосу үшін қажет.
5. **WIDTH, HEIGHT = 600, 400** - Ойын терезесінің ені мен биіктігін анықтау. Бұл мәндер экранның өлшемдерін береді (600x400 пиксель).
6. **screen = pygame.display.set\_mode((WIDTH, HEIGHT))** - Экранды құру. WIDTH пен HEIGHT параметрлері арқылы экран терезесін құрады.
7. **pygame.display.set\_caption("Memory Game")** - Ойын терезесінің тақырыбын орнату. Бұл код "Memory Game" деген мәтінді терезенің жоғарғы жағында көрсетеді.
8. **BLACK = (0, 0, 0)**, **WHITE = (255, 255, 255)**, **GRAY = (169, 169, 169)** - Түстерді анықтау. RGB форматында қара, ақ және сұр түстерді анықтау.
9. **circle\_radius = 30** - Шеңбердің радиусын анықтау, яғни ойынның әр шеңберінің диаметрі 60 пиксель болады.
10. **circle\_count = 8** - Ойындағы жалпы шеңберлердің саны. Бұл 8 шеңбер, бірақ олар екі рет қайталанатын жұптар болады.
11. **pairs = circle\_count // 2** - Жұптардың санын анықтау. 8 шеңбердің 4 жұп болатынын көрсетеді.
12. **colors = [pygame.Color(random.choice(['red', 'blue', 'green', 'yellow', 'purple', 'orange', 'pink', 'cyan'])) for \_ in range(pairs)]** - Жұптар үшін түстерді кездейсоқ таңдау. pygame.Color() арқылы кездейсоқ түстер генерацияланады.
13. **colors \*= 2** - Түстерді екі рет қайталау. Бұл код шеңберлердің әрқайсысының түстерін екі рет қайталайды, яғни әр түс жұп болып қайталанады.
14. **random.shuffle(colors)** - Түстерді кездейсоқ ретке келтіру. Бұл код барлық шеңберлердің түстерін араластырып, оларды кездейсоқ орналастырады.
15. **rows, cols = 4, 4** - Шеңберлердің орналасу торын анықтау. Бұл 4 жол мен 4 бағаннан тұратын торды білдіреді.
16. **positions = []** - Шеңберлердің орнын сақтау үшін бос тізім жасау.
17. **for i in range(rows):** және **for j in range(cols):** - Шеңберлердің экранда орналасуын анықтау үшін екі цикл. Әр циклда шеңбердің орны анықталады.
18. **x = j \* (WIDTH // cols) + (WIDTH // cols) // 2** және **y = i \* (HEIGHT // rows) + (HEIGHT // rows) // 2** - Шеңбердің координаттарын есептеу. Әр шеңбер үшін оның (x, y) орнын тордағы бағандар мен жолдар бойынша есептейді.
19. **guessed = [False] \* circle\_count** - Тапқан шеңберлердің жағдайын бақылау. Егер пайдаланушы шеңберді тапса, сол шеңбердің индексі True болады.
20. **first\_guess = None** және **second\_guess = None** - Бірінші және екінші таңдалған шеңберлерді сақтау үшін айнымалылар. Бұл айнымалылар шеңберлерді салыстыру үшін қолданылады.
21. **score = 0** - Ұпайды сақтау. Әр дұрыс жұп үшін ұпай саны артады.
22. **running = True** - Ойынның жұмыс істеп жатқанын көрсету.
23. **game\_phase = 'show'** - Ойынның фазасын бақылау. Алдымен шеңберлер көрсетіледі, кейін жасырылады.
24. **show\_time = 3** - Шеңберлерді көрсету үшін уақыт. Бұл 3 секунд болғандықтан, ойыншы шеңберлерді есте сақтай алады.
25. **while running:** - Ойынның негізгі циклі. Бұл цикл ойынның жұмысын және пайдаланушының әрекеттерін бақылап отырады.
26. **screen.fill(BLACK)** - Экранды қара түспен толтыру. Бұл әр жаңа кадрда экранды тазалауға көмектеседі.
27. **for event in pygame.event.get():** - Пайдаланушының әрекеттерін тексеру. Оған шертулер, терезеден шығу әрекеттері және т.б. жатады.
28. **if event.type == pygame.QUIT:** - Егер ойын терезесінен шығу әрекеті жасалса, ойынның аяқталуы.
29. **if event.type == pygame.MOUSEBUTTONDOWN:** - Егер пайдаланушы тышқанмен шертсе, оны өңдеу.
30. **for i, (x, y) in enumerate(positions):** - Әр шеңбердің орнын тексеру. Пайдаланушы қай шеңберді таңдағанын анықтау.
31. **if not guessed[i]:** - Егер шеңбер дұрыс табылмаса, оны қарастыру.
32. **if (x - circle\_radius <= mouse\_x <= x + circle\_radius) and (y - circle\_radius <= mouse\_y <= y + circle\_radius):** - Егер тышқан шеңбердің үстінде болса, оны анықтау.
33. **if first\_guess is None:** - Егер бірінші шеңбер таңдалмаса, оны таңдау.
34. **elif second\_guess is None:** - Егер екінші шеңбер таңдалмаса, оны таңдау.
35. **if colors[first\_guess] == colors[second\_guess]:** - Егер екі шеңбердің түстері бірдей болса, оларды дұрыс тапқан деп санау.
36. **guessed[first\_guess] = guessed[second\_guess] = True** - Дұрыс тапқан шеңберлерді белгілеу.
37. **score += 1** - Дұрыс жұп үшін ұпай қосу.
38. **first\_guess = None** және **second\_guess = None** - Бірінші және екінші таңдауларды қайтадан босату.
39. **if game\_phase == 'show':** - Егер ойын фазасы шеңберлерді көрсету болса, шеңберлерді көрсету.
40. **time.sleep(show\_time)** - Шеңберлерді белгілі бір уақыт көрсеткеннен кейін жасыру үшін ұйқы режимі.
41. **game\_phase = 'hide'** - Ойын фазасын өзгерту. Егер шеңберлер көрсетілген болса, оларды жасыру.
42. **if game\_phase == 'hide':** - Егер ойын фазасы шеңберлерді жасыру болса, оларды жасыру.
43. **pygame.draw.circle(screen, color, (x, y), circle\_radius)** - Шеңберді экранға салу. Шеңбердің түсі мен орны беріледі.
44. **font = pygame.font.Font(None, 36)** - Ұпай көрсетілген мәтіннің шрифтін жасау.
45. **score\_text = font.render(f"Score: {score}", True, WHITE)** - Ұпайды экранға шығару үшін мәтінді жасау.
46. **screen.blit(score\_text, (10, 10))** - Ұпай мәтінін экранда көрсетілген орынға орналастыру.
47. **pygame.display.flip()** - Экранды жаңарту.
48. **pygame.time.Clock().tick(30)** - FPS-ті орнату. Бұл әр секундта 30 кадрға дейін жаңартуға мүмкіндік береді.
49. **pygame.quit()** - Pygame кітапханасын жабу.

Бұл түсіндірмелердің көмегімен кодтың әрбір бөлігін тереңірек түсінуге болады.