



алгоритмика

# PYTHON

Занятие 24

# Сегодня на уроке

---

- Взаимодействие спрайтов
- Артефакты
- Усложнение игры



# Вопросы

---

1. Что такое слой?
2. Какой спрайт будет на переднем плане, первый или последний?
3. Как различать спрайты одного класса?
4. Как отследить взаимодействие спрайтов?
5. Для чего используется переменная `game_over` в игре?
6. Какие действия останавливались, если `game_over=0`?
7. Откуда начинается отсчет координат в окне приложения?
8. Что такое случайное значение?

# Размещение спрайта в игровом окне

---

Создать объект класса в указанном месте

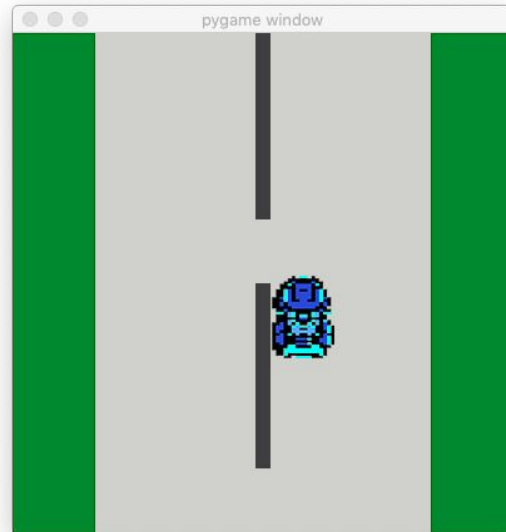
1 `car1 = Car(randint(25,375), 'Car1.png')`

Задать координату у для левого верхнего угла

2 `car1.rect.y=0`

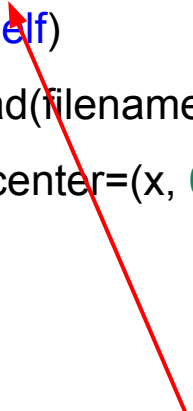
Разместить объект сверху прямоугольника по размеру загруженной картинки

3 `sc.blit(car1.image, car1.rect)`



# Добавление группы

```
class Game_sprite(pygame.sprite.Sprite):  
    def __init__(self, x, filename, group):  
        pygame.sprite.Sprite.__init__(self)  
        self.image = pygame.image.load(filename).convert_alpha()  
        self.rect = self.image.get_rect(center=(x, 0))  
        self.add(group)  
  
car1 = Game_sprite(100, 'Car1.png', user_car)
```



Группа для спрайта будет определена в момент создания объекта класса

# Взаимодействие спрайтов

---

Метод **`spritecollideany()`** проверяет взаимодействие конкретного спрайта с любым из спрайтов из группы. Функция принимает первым аргументом спрайт, который проверяется, вторым – группу.

```
pygame.sprite.spritecollideany(имя_спрайта, имя_группы_взаимодействия)
```

Проверяет взаимодействие гоночной машины и противника

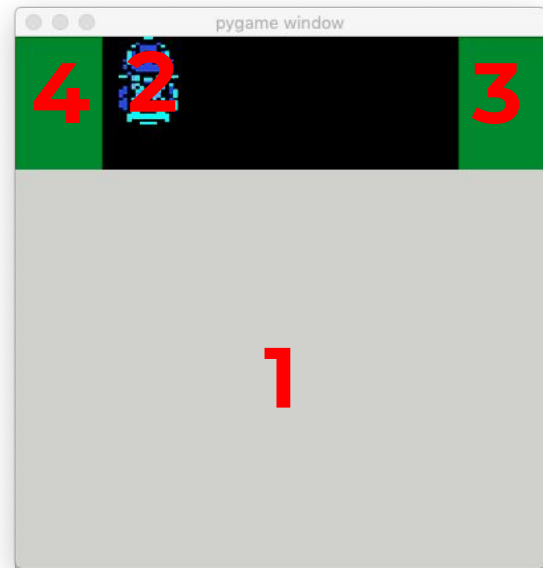
```
pygame.sprite.spritecollideany(car1, cars)
```



# Слои на игровом поле

Каждый **графический объект** – это **слой**. Порядок слоев определяется их созданием в программе. **Чем позже** создан слой, **тем выше** его положение.

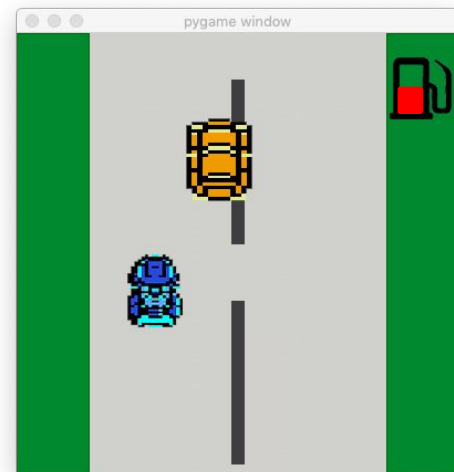
- 4 `sc.blit(gr_left.image, gr_left.rect)`
- 3 `sc.blit(gr_right.image, gr_right.rect)`
- 2 `sc.blit(car1.image, car1.rect)`
- 1 `sc.blit(road.image, road.rect)`



# Вопрос

---

Когда начинается игровой процесс?





# Вопрос-ответ

---

Когда начинается игровой процесс?

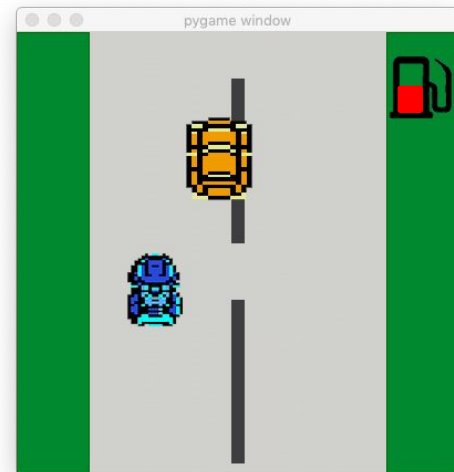
Сразу после запуска  
программы



# Вопрос

---

Что должно появляться в игре до начала игрового процесса?



# Вопрос-ответ

---

Что должно появляться в игре до начала игрового процесса?

Меню игры



# Меню игры

---

Меню игры позволяет контролировать запуск и остановку игры. Действия регулируются специальными кнопками.



Запуск игры



Закрытие приложения



# Задание

---

Для того, чтобы можно было настроить меню, на время остановим запуск игры. Измените указатель на значение `game_over` с 1 на 0.

```
game=True  
game_over = 1 → 0
```

```
while game:  
    keys = pygame.key.get_pressed()  
    ...
```



# Задание. Решение

---

Для того, чтобы можно было настроить меню, на время остановим запуск игры. Измените указатель на значение `game_over` с 1 на 0.

```
game=True
game_over = 0

while game:
    keys = pygame.key.get_pressed()
    ...
```



# Задание

---

Скачайте файлы: menu.png, btn\_play.png, btn\_exit.png



**menu.png**



**btn\_play.png**



**btn\_exit.png**

# Задание

Создайте спрайты с этими изображениями и задайте соответствующие группы. Разместите кнопки, как показано на рисунке.



menu / menu\_group



btn\_start / play\_group



btn\_stop / stop\_group





# Задание. Решение

Создайте спрайты с этими изображениями и задайте соответствующие группы. Разместите кнопки, как показано на рисунке.

```
menu_group = pygame.sprite.Group()
play_group = pygame.sprite.Group()
stop_group = pygame.sprite.Group()
```

```
menu=Game_sprite(200,'menu.png',menu_group)
btn_start = Game_sprite(200,'btn_play.png',play_group)
btn_stop = Game_sprite(200,'btn_exit.png',stop_group)
```

```
-----
menu.rect.y = 0
btn_start.rect.y=150
btn_stop.rect.y=250
-----
```

```
sc.blit(menu.image,menu.rect)
play = sc.blit(btn_start.image,btn_start.rect)
stop = sc.blit(btn_stop.image,btn_stop.rect)
```



# Вопрос

---

Как пользователь будет выбирать пункт меню?



# Вопрос-ответ

---

Как пользователь будет выбирать пункт меню?

Нажимать левой  
кнопкой мыши на  
кнопку



# Вопрос

---

Как мы отслеживаем нажатие кнопки на клавиатуре?



# Вопрос-ответ

---

Как мы отслеживаем нажатие кнопки на клавиатуре?

```
keys = pygame.key.get_pressed()
```

```
if keys[pygame.K_LEFT]:
```

```
    car1.rect.x = car1.rect.x - 3
```



# Отслеживание событий мыши

Метод **MOUSEBUTTONDOWN** класса `pygame` позволяет отследить нажатие клавиши на мышке.

```
if i.type==pygame.MOUSEBUTTONDOWN:
```

```
    if i.button==1:
```

```
        ...
```

i.button	Клавиша мыши
1	Левая
2	Средняя (колесико)
3	Правая

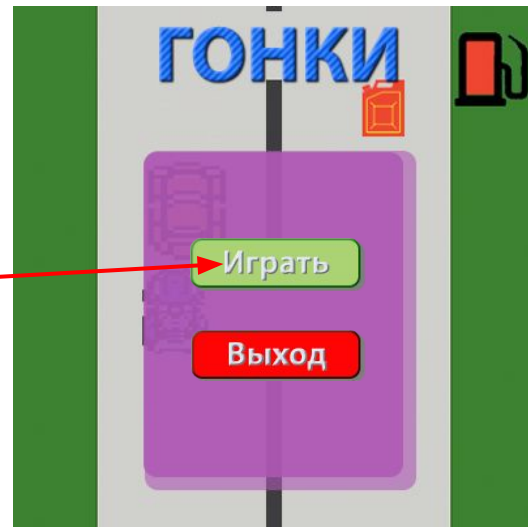


# Отслеживание событий мыши

Метод **pos** объекта событий игры возвращает точку, в которой было совершено действие.

```
pos=i.pos
```

```
print(pos) → (250, 170)
```



# Задание

---

Добавьте в бесконечный цикл проверку нажатия левой клавиши мыши. После нажатие левой кнопки, в консоль выведите координаты нажатия.

```
if i.type == pygame.QUIT:  
    game=False  
  
if i.type==pygame.MOUSEBUTTONDOWN:  
    if i.button==1:  
        pos=i.pos  
        print(pos)
```





# Связь мыши и спрайта

Метод **MOUSEBUTTONDOWN** класса `pygame` позволяет отследить нажатие клавиши на мышке.

```
if i.type==pygame.MOUSEBUTTONDOWN:
```

```
    if i.button==1:
```

```
        ...
```

i.button	Клавиша мыши
1	Левая
2	Средняя (колесико)
3	Правая



# Связь мыши и спрайта

Для того, чтобы программа смогла понять, нажал ли пользователь в границах спрайта, надо:

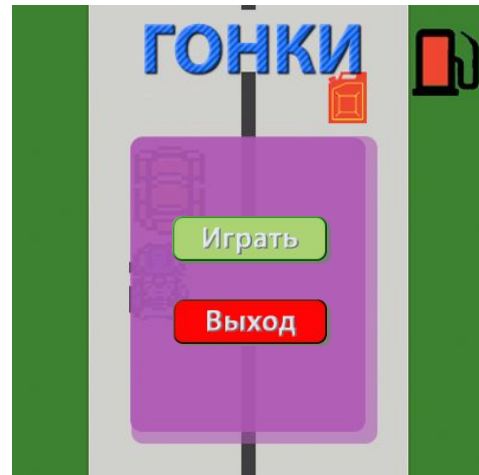
1. Создать указатель на прямоугольник, в который помещен спрайт;

```
play=sc.blit(btn_start.image,btn_start.rect)
```

2. Вызвать метод **collidepoint** у этого объекта.

```
play.collidepoint(pos)
```

Метод **collidepoint()** проверяет, находится ли точка, координаты которой были переданы в качестве аргумента, в пределах прямоугольника, к которому применяется метод. **Результат работы True или False.**



# Задание

---

Перед бесконечным циклом добавьте создание переменной-указателя на прямоугольник вокруг кнопки «Старт» из меню



# Задание. Решение

---

Перед бесконечным циклом добавьте создание переменной-указателя на прямоугольник вокруг кнопки «Старт» из меню

```
play=sc.blit(btn_start.image,btn_start.rect)
```

```
pygame.display.update()
```

```
while game:
```

```
...
```



# Вопрос

---

Что должно произойти, если пользователь нажмет на кнопку «Играть»?



# Вопрос-ответ

---

Что должно произойти, если пользователь нажмет на кнопку «Играть»?

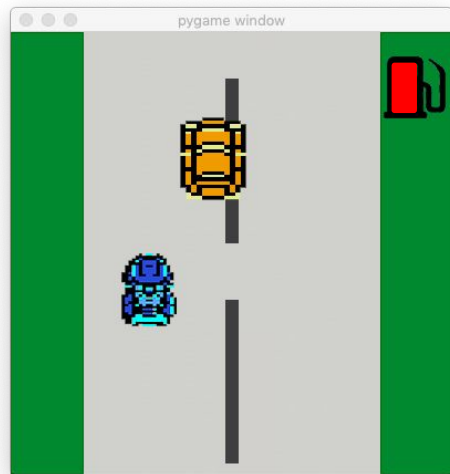
Начаться игра.  
Все герои должны  
занять свои места



# Вопрос

---

Как выглядит игровое окно в момент запуска игры?



# Вопрос

---

Как выглядит игровое окно в момент запуска игры?

Сверху начинают  
двигаться противники,  
бак полный

`car2.rect.y = 0`

`k=14`

`f=42`

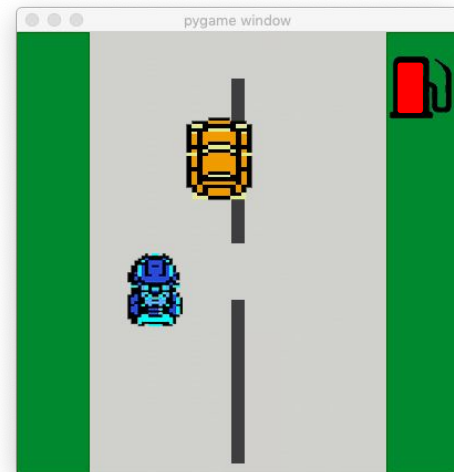




# Вопрос

---

Какая переменная отвечает за запуск программы?



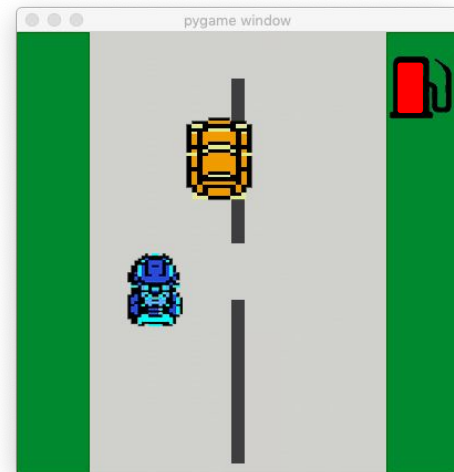
# Вопрос-ответ

---

Какая переменная отвечает за запуск программы?

`game_over`

`game_over=1`



# Задание

Добавьте в программу проверку нажатия на кнопку «Старт» в созданном меню

```
if i.type == pygame.QUIT:  
    game=False
```

```
if i.type == pygame.MOUSEBUTTONDOWN:  
    if i.button==1:  
        pos=i.pos  
        if play.collidepoint(pos):  
            game_over=1  
            car2.rect.y = 0  
            k=14  
            f=42  
            pygame.display.update()
```



# Вопрос

---

Что должно произойти, если пользователь нажмет кнопку «Выход» в меню?



# Вопрос-ответ

---

Что должно произойти, если пользователь нажмет кнопку «Выход» в меню?

Игровое окно  
должно закрыться



# Вопрос

---

В какой еще момент игровое окно закрывается?



# Вопрос

---

В какой еще момент игровое окно закрывается?

Если пользователь  
нажимает кнопку «Заккрыть»  
у игрового окна

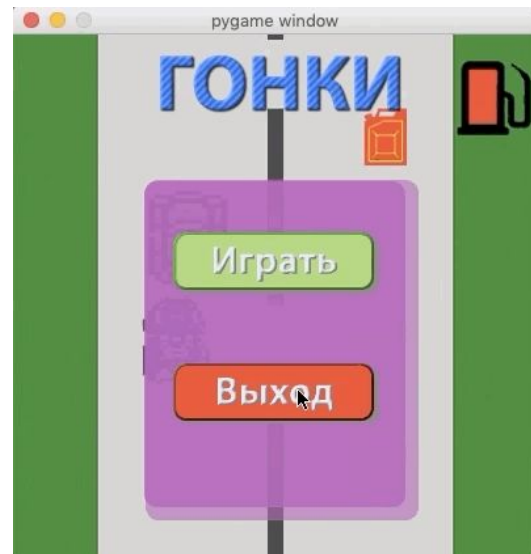
```
for i in pygame.event.get():  
    if i.type == pygame.QUIT:  
        game=False
```



# Задание

---

Добавьте проверку нажатия на кнопку «Выход». Если пользователь нажал на кнопку «Выход», игровое окно должно закрыться.





# Задание. Решение

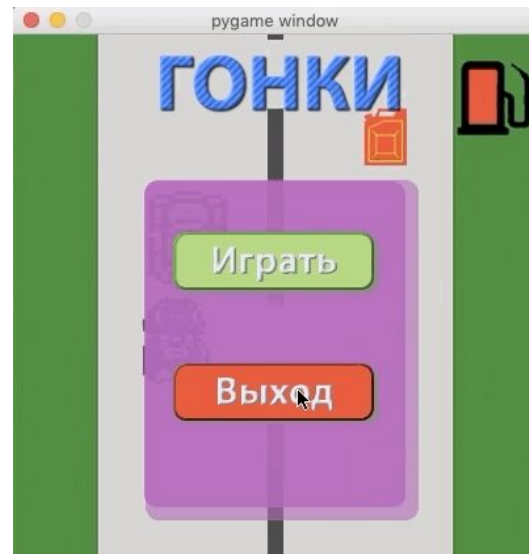
Добавьте проверку нажатия на кнопку «Выход». Если пользователь нажал на кнопку «Выход», игровое окно должно закрыться.

```
play=sc.blit(btn_start.image,btn_start.rect)
stop = sc.blit(btn_stop.image,btn_stop.rect)
```

```
...
```

```
if i.type==pygame.MOUSEBUTTONDOWN:
    if i.button==1:
        pos=i.pos
        if play.collidepoint(pos):
            game_over=1
            car2.rect.y = 0
            k=14
            f=42
            pygame.display.update()
```

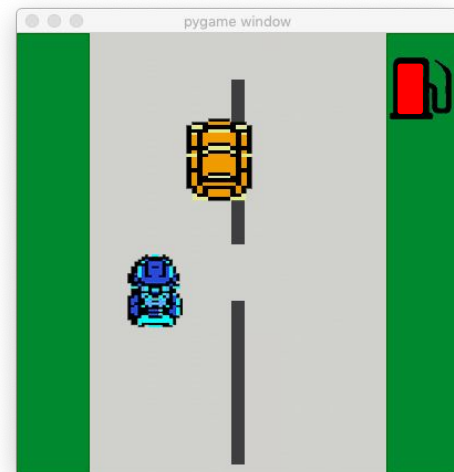
```
elif stop.collidepoint(pos):
    game=False
```



# Вопрос

---

Когда заканчивается игра?

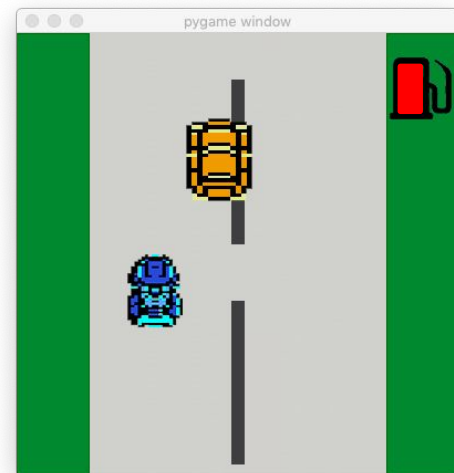


# Вопрос-ответ

---

Когда заканчивается игра?

Когда происходит  
столкновение или  
заканчивается топливо



# Вопрос

---

Как происходит перезапуск игры?



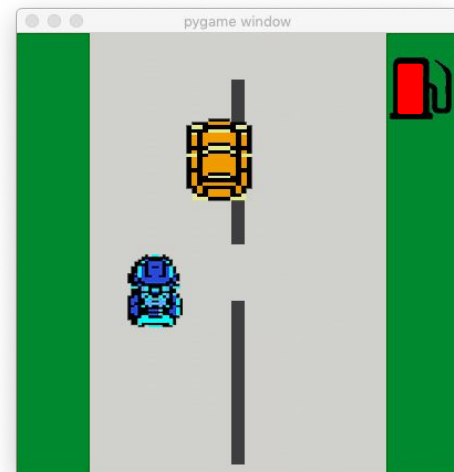
# Вопрос

---

Как происходит перезапуск игры?

Нажимает 9 на  
клавиатуре и игра  
начинается заново

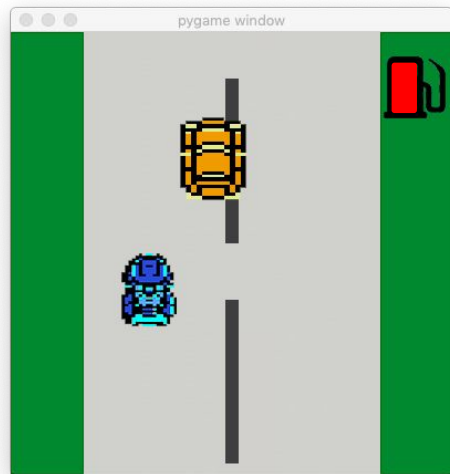
```
elif keys[pygame.K_9]:  
    game_over=1  
    car2.rect.y = 0  
    k=14  
    f=42
```



# Вопрос

---

Что должно сейчас появляться после окончания игры?



# Вопрос-ответ

---

Что должно сейчас появляться после окончания игры?

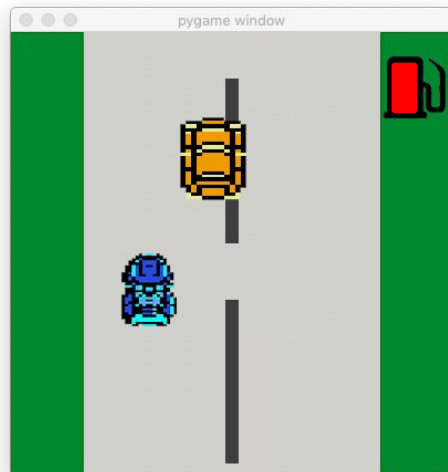
Меню



# Вопрос

---

Какую конструкцию мы используем, чтобы несколько раз выполнить одно и тоже, не дописывая при этом код?





# Вопрос

---

Какую конструкцию мы используем, чтобы несколько раз выполнить одно и тоже, не дописывая при этом код?

Функцию



# Задание

---

## Удалите перезапуск программы через 9.

Создайте функцию `menu_def()` со следующим телом:

```
sc.blit(menu.image,menu.rect)
play = sc.blit(btn_start.image,btn_start.rect)
stop = sc.blit(btn_stop.image,btn_stop.rect)
pygame.display.update()
```

**Функция должна быть расположена ниже класса спрайта**

# Задание. Решение

---

Удалите перезапуск программы через 9.  
Создайте функцию `menu_def()` со следующим телом:

```
def menu_def():  
    sc.blit(menu.image, menu.rect)  
    play = sc.blit(btn_start.image, btn_start.rect)  
    stop = sc.blit(btn_stop.image, btn_stop.rect)  
    pygame.display.update()
```

**Функция должна быть расположена ниже класса  
спрайта**



# Вызов функции

---

Вызов функции может происходить после определенных команд, а не после конкретного действия пользователя. В этом случае вызов функции выглядит, как:

**имя\_функции()**

```
if pygame.sprite.spritecollideany(car1, cars):  
    print("Авария!!!")  
    game_over=0  
    menu_def()  
  
if f<0:  
    game_over=0  
    menu_def()
```

# Вопросы

---

1. Как проверить нажатие клавиши мыши?
2. Как отследить взаимодействие спрайта и мыши?
3. Что является условием окончания игры?
4. Что такое перезапуск игры?
5. Как вызвать функцию после определенной команды в программе?

# Задание на дом. Уровень 1

---

Для появления меню после проигрыша добавьте задержку с помощью метода `time.delay()`

Функция `delay()` принимает количество миллисекунд ( $1000 \text{ мс} = 1 \text{ с}$ ). Если передано значение 20, то за секунду экран обновится 50 раз. Другими словами, частота составит 50 кадров в секунду.

Пример: `pygame.time.delay(20)`



# Задание на дом. Уровень 2

---

Для появления меню после проигрыша добавьте задержку с помощью метода `time.delay()`.

Добавьте подсчет количества противников, которых удалось избежать машине до столкновения или окончания топлива. После завершения игры, выведите эту информацию в консоль.

