Titel: Komet dræberen

Spiloplevelse:

Det forgår i det ydre rum. Du flyver rundt i et rumskib. Der er kometer som flyver på kryds og tværs. Du skal skyde så mange som muligt ned inden en af kometerne rammer dig og spillet er ovre.

Elementer i spillet:

- Stjerne baggrund
- Rumskib
- En eller flere kometer
- Skud (flere skud samtidig)
- Point system

Del mål:

1. Sprites til spillet
2. Spil vindue med stjernefyldt baggrund
3. Rumskib der kan roter og flyve fremad
4. Rumskibet kan skyde
5. Kometer der kommer fra alle side med forskelligt tids interval
6. Komet ødelægges når den rammes med et skud
7. Point system

Del mål 1 - Sprites til spillet

Der skal bruges sprites til følgende

- rumskib
- skud
- lille meteor
- stor meteor

Gå til https://kenney.nl/assets/simple-space og download pakken

Find i pakken følgende filer og omdøb dem til navn i ():

ship_F.png	(ship.png)
star_small.png	(bullet.png)
meteor_detailedLarge.png	(meteor_large.png)
meteor_detailedSmall.png	(meteor_small.png)

Opret en mappe til dit spil – kald mappen komet_dræber)

opret i denne mappe en ny mappe – kald mappen images

placer de fire png filer i mappen

Del mål 2 - Spil vindue med stjernefyldt baggrund

Vi definer spil vinduets størrelse som også bliver rammen for området hvor i stjerner placers.

Opret en fil kaldet main.py i folderen komet_dræber

Tilføj denne kode til main.py

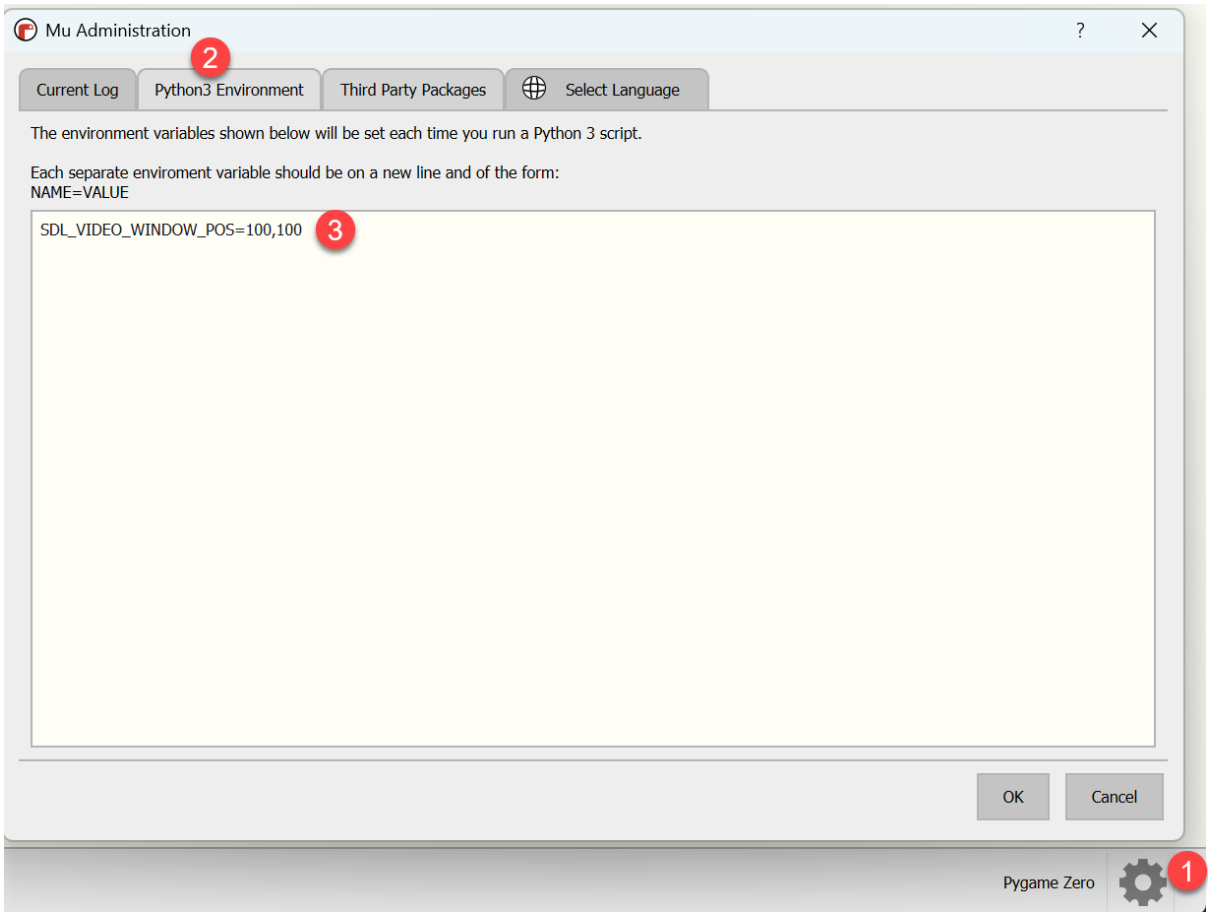
```
#Komet dræber spil - skyd kometerne ned inden de rammer dig!

WIDTH = 600
HEIGHT = 800

def draw():
    pass
```

Tilpas størrelsen efter din skærms opløsning
Afprøv koden (husk at sætte mode til pygame zero

Det kan være nødvendigt at ændre dit spil vindues start position med denne kode
SDL_VIDEO_WINDOW_POS=100,100



Hvis det virker som det skal, så lad os tilføje nogle stjerner til baggrunden

Vi vil have stjernerne til at placer sig tilfældigt så vi skal bruge random biblioteket

```
#Komet dræber spil - skyd kometerne ned inden de rammer dig!
import random

WIDTH = 600
HEIGHT = 800
```

Stjerner skal placeres på tilfældige x og y koordinater og skal have forskellige størrelser.
vi starter med at generer disse parameter for 100 stjerner, som vi placer i et array kaldet stars

```
# Stjerner
stars = []
for _ in range(100):
    stars.append({
        "x": random.randint(0, WIDTH),
        "y": random.randint(0, HEIGHT),
        "size": random.choice([1, 2]),

    })
```

Der efter vil vi have dem tegnet på skærmen så vi tilføjer følgende til draw()

```
def draw():
    # Tegn stjerner
    for star in stars:
        screen.draw.filled_circle((star["x"], star["y"]), star["size"], "white")
```

husk at slette linjen med ”pass”

prøv nu spillet af.
ændre lidt på antal og størrelsesintervallet og se resultatet

Del mål 3 - Rumskib der kan roter og flyve fremad

Vi starter med at definer vores spillers rumskib objekt. Fremadrettet vil vi referer til det som ship

Skibet er en actor og grafiken er vores ship.png fil

Skibet skal til at starte med være placeres midt på skærmen

Tilføj denne kode til starten programmet lige efter HEIGHT linjen

```
7  ship = Actor('ship')
8  ship.pos = (WIDTH // 2, HEIGHT // 2)
```

Hvis du undere dig over // i stedet for en enkelt /
så er det en indbygget funktion der sikre at resultatet er et helt tal eks

5 / 2 = 2.5 mens 5 // 2 = 2

Opdater draw() med følgende kode

```
26      # Tegn Spiller
27      ship.draw()
```

Test at rumskibet nu vises på skærmen.

Prøve at eksperimenter med forskellen ved at have ship.draw() koden før / efter koden der tegner stjernerne på baggrunden. Kan du se forskellen (hint: sætter stjerne antal op og skift farven)

Nu skal vi have rumskibet til at roter når man bruger venstre og højre piletaster

Til det skal vi bruge en angle parameter på vores ship objekt.

Tilføj denne kode

```
7  # Spiller
8  ship = Actor('ship')
9  ship.pos = (WIDTH // 2, HEIGHT // 2)
10 ship.angle = 0
```

Og tilføj en ny update() funktion med følgende kode

```
31 def update():
32     # Roter spiller
33     if keyboard.left:
34         ship.angle += 4
35     if keyboard.right:
36         ship.angle -= 4
```

Prøv det af og se om det virker efter hensigten?

Det gør det ikke

Skibet roter, men det gentegner sig bare oven i sig selv.
Det er fordi vi hver gang draw() kaldes, skal huske at rense spil vinduet.

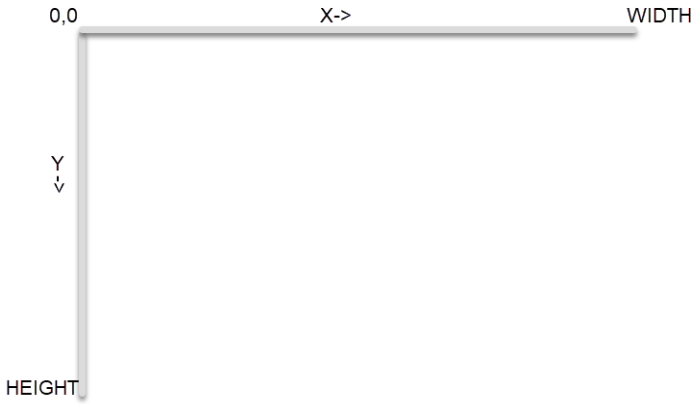
Tilføj denne kode som det første når draw() kaldes

```
23 def draw():
24     # Rens vinduet
25     screen.clear()
26
```

Prøv om det virker bedre nu

Nu skal vi have rumskibet til at kunne flyve fremad

Vi kender koordinat systemet i spillet



Det kan virke lige til at bare ændre ship.y koordinat når der trykkes på piltast op

Prøv at indsætte denne linje i update()

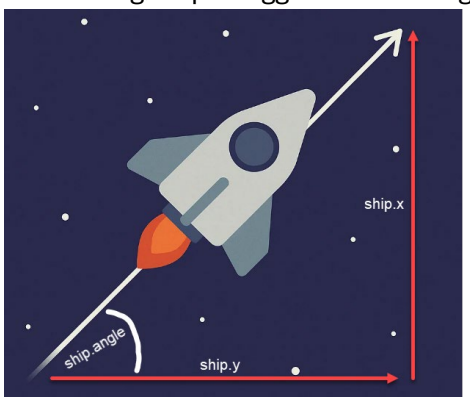
```
42     # Bevæg spiller fremad
43     if keyboard.up:
44         ship.y -= 4
```

Se hvad der sker.

Det virker fint i fremadretning, men hvad nu når skibet er roteret?

Fordi skibet kan roter bliver vi nød til at nedbryde fremad bevægelsen til hvor meget bevægelsen påvirker skibets x og y værdi

det kan vi gøre på baggrund af retning (ship.angle) og lidt matematik



Ændret fremad koden du lige sat ind til det her

```
42      # Bevæg spiller fremad
43      if keyboard.up:
44          rad = math.radians(ship.angle)
45          ship.x += -math.sin(rad) * 1
46          ship.y -= math.cos(rad) * 1
```

Og da vi bruger math biblioteket skal vi have det sat ind i toppen af filen

```
2   import random
3   import math
```

Prøv nu spillet ad. Er det bedre?

Du kan nok regne ud hvordan du ændre hastigheden på rumskibet!

Når, men i sådan et rumskib spil skal rumskibet jo ikke bare stoppe når man slipper pil op tasten. Så vi skal have skibet til at få fart på og lige så stille bremse op.

vi skal ændre lidt på vores kode fra før så vi gemmer vores x y ændring i nogle nye parametre vx og vy

```
43      # Bevæg spiller fremad
44      if keyboard.up:
45          rad = math.radians(ship.angle)
46          ship.vx += -math.sin(rad) * 1
47          ship.vy -= math.cos(rad) * 1
```

De parametre skal vi tilføje vores ship objekt

```
8   # Spiller
9   ship = Actor('ship')
10  ship.pos = (WIDTH // 2, HEIGHT // 2)
11  ship.angle = 0
12  ship.vx = 0
13  ship.vy = 0
```

I update() indsættes denne nye kode del.

```
43      # Bevæg spiller fremad
44      if keyboard.up:
45          rad = math.radians(ship.angle)
46          ship.vx += -math.sin(rad) * 1
47          ship.vy -= math.cos(rad) * 1
48
49      # Friktion og position
50      ship.x += ship.vx
51      ship.y += ship.vy
52      ship.vx *= 0.98
53      ship.vy *= 0.98
```

bemærk der kun er et enkelt indryk.
det vil sige at fremad handling kun registreres når der trykkes på pil op.
bagefter bliver ændring lagt ind på skibets position.
derefter sker der en reduktion af vx og vy hver gang update() kaldes. (hvilket pygame zero gør automatisk)
Det giver bremse effekten.

Prøv nu spillet

Flyver rumskibet lidt hurtigt? Prøv dig lidt frem med forskellig hastighedsværdier.

Det er lidt irriterende skibet flyver uden for skærm området.
Lad os hurtig fikse det så rumskibet kommer ud over vindues grænsen flyttes til modsatte kant.

Tilføj denne kode til sidst i update()

```
55      # Skærmgrænser (wrap around)
56      if ship.x < 0:
57          ship.x = WIDTH
58      if ship.x > WIDTH:
59          ship.x = 0
60      if ship.y < 0:
61          ship.y = HEIGHT
62      if ship.y > HEIGHT:
63          ship.y = 0
```

Prøv det af