# Snake v jazyku Python - Používanie modulu Pygame

<https://www.geeksforgeeks.org/snake-game-in-python-using-pygame-module/>

Hra Snake je jednou z najpopulárnejších arkádových hier všetkých čias. V tejto hre je hlavným cieľom hráča chytiť maximálny počet plodov bez toho, aby narazil do steny alebo do seba. Vytvorenie hry s hadom možno brať ako výzvu pri učení sa Pythonu alebo Pygame. Je to jeden z najlepších projektov vhodných pre začiatočníkov, ktorý by mal každý začínajúci programátor brať ako výzvu. Naučiť sa vytvárať videohru je zaujímavé a zábavné učenie.

Na vytvorenie tejto hadej hry použijeme **[Pygame](https://www.geeksforgeeks.org/introduction-to-pygame/)**.**Pygame** je knižnica s otvoreným zdrojovým kódom, ktorá je určená na tvorbu videohier. Má vstavanú grafickú a zvukovú knižnicu. Je tiež vhodný pre začiatočníkov a multiplatformový.

## Inštalácia.

Ak chcete nainštalovať Pygame, musíte otvoriť terminál alebo príkazový riadok a zadať nasledujúci príkaz:

pip install pygame

Po nainštalovaní Pygame sme pripravení vytvoriť našu skvelú hadiu hru.

**Podrobný prístup k vytvoreniu hry Snake pomocou Pygame:**

**Krok 1:**Najprv importujeme potrebné knižnice.

* Potom definujeme šírku a výšku okna, v ktorom sa bude hra hrať.
* A definujte farbu vo formáte RGB, ktorú použijeme v našej hre na zobrazenie textu.

# importing libraries

import pygame

import time

import random

snake\_speed = 15

# Window size

window\_x = 720

window\_y = 480

# defining colors

black = pygame.Color(0, 0, 0)

white = pygame.Color(255, 255, 255)

red = pygame.Color(255, 0, 0)

green = pygame.Color(0, 255, 0)

blue = pygame.Color(0, 0, 255)

**Krok 2:** Po importe knižníc musíme inicializovať Pygame pomocou metódy **pygame.init().**

* Vytvorte herné okno pomocou šírky a výšky definovanej v predchádzajúcom kroku.
* Tu **sa pygame.time.Clock()** bude ďalej používať v hlavnej logike hry na zmenu rýchlosti hada.

# Initialising pygame

pygame.init()

# Initialise game window

pygame.display.set\_caption('GeeksforGeeks Snakes')

game\_window = pygame.display.set\_mode((window\_x, window\_y))

# FPS (frames per second) controller

fps = pygame.time.Clock()

**Krok 3:**Inicializujte polohu hada a jeho veľkosť.

* Po inicializácii polohy hada inicializujte polohu ovocia náhodne kdekoľvek v definovanej výške a šírke.
* Nastavením smeru na RIGHT zabezpečíme, že vždy, keď používateľ spustí program/hru, had sa musí presunúť priamo na obrazovku.

# defining snake default position

snake\_position = [100, 50]

# defining first 4 blocks of snake

# body

snake\_body = [ [100, 50],

[90, 50],

[80, 50],

[70, 50]

]

# fruit position

fruit\_position = [random.randrange(1, (window\_x//10)) \* 10,

random.randrange(1, (window\_y//10)) \* 10]

fruit\_spawn = True

# setting default snake direction

# towards right

direction = 'RIGHT'

change\_to = direction

**Krok 4:**Vytvorte funkciu na zobrazenie skóre hráča.

* V tejto funkcii najprv vytvoríme objekt písma, t. j. farba písma pôjde sem.
* Potom použijeme render na vytvorenie povrchu pozadia, ktorý zmeníme vždy, keď sa naše skóre aktualizuje.
* Vytvorenie obdĺžnikového objektu pre objekt povrchu textu (kde sa text obnoví)
* Potom zobrazíme naše skóre pomocou **blesku. blit** berie dva argumenty**screen.blit(pozadie,(x,y))**

# initial score

score = 0

# displaying Score function

def show\_score(choice, color, font, size):

# creating font object score\_font

score\_font = pygame.font.SysFont(font, size)

# create the display surface object

# score\_surface

score\_surface = score\_font.render('Score : ' + str(score), True, color)

# create a rectangular object for the

# text surface object

score\_rect = score\_surface.get\_rect()

# displaying text

game\_window.blit(score\_surface, score\_rect)

**Krok 5**: Teraz vytvorte funkciu game over, ktorá bude predstavovať skóre po tom, čo hada zasiahne stena alebo sám seba.

* V prvom riadku vytvárame objekt písma na zobrazenie notových zápisov.
* Potom vytvárame textové povrchy na vykresľovanie skóre.
* Potom nastavíme polohu textu v strede hrateľnej oblasti.
* Zobrazenie skóre pomocou **blit** a aktualizácia notového zápisu aktualizáciou povrchu pomocou flip().
* Používame sleep(2) na počkanie 2 sekúnd pred zatvorením okna pomocou quit().

# game over function

def game\_over():

# creating font object my\_font

my\_font = pygame.font.SysFont('times new roman', 50)

# creating a text surface on which text

# will be drawn

game\_over\_surface = my\_font.render('Your Score is : ' + str(score), True, red)

# create a rectangular object for the text

# surface object

game\_over\_rect = game\_over\_surface.get\_rect()

# setting position of the text

game\_over\_rect.midtop = (window\_x/2, window\_y/4)

# blit will draw the text on screen

game\_window.blit(game\_over\_surface, game\_over\_rect)

pygame.display.flip()

# after 2 seconds we will quit the

# program

time.sleep(2)

# deactivating pygame library

pygame.quit()

# quit the program

quit()

**Step 6:** Now we will be creating our main function that will do the following things:

* We will be validating the keys that will be responsible for the movement of the snake, then we will be creating a special condition that the snake should not be allowed to move in the opposite direction instantaneously.
* After that, if snake and fruit collide we will be incrementing the score by 10 and new fruit will be spanned.
* After that, we are checking that is the snake hit with a wall or not. If a snake hits a wall we will call game over function.
* If the snake hits itself, the game over function will be called.
* And in the end, we will be displaying the scores using the show\_score function created earlier.

# Main Function

while True:

# handling key events

for event in pygame.event.get():

if event.type == pygame.KEYDOWN:

if event.key == pygame.K\_UP:

change\_to = 'UP'

if event.key == pygame.K\_DOWN:

change\_to = 'DOWN'

if event.key == pygame.K\_LEFT:

change\_to = 'LEFT'

if event.key == pygame.K\_RIGHT:

change\_to = 'RIGHT'

# If two keys pressed simultaneously

# we don't want snake to move into two directions

# simultaneously

if change\_to == 'UP' and direction != 'DOWN':

direction = 'UP'

if change\_to == 'DOWN' and direction != 'UP':

direction = 'DOWN'

if change\_to == 'LEFT' and direction != 'RIGHT':

direction = 'LEFT'

if change\_to == 'RIGHT' and direction != 'LEFT':

direction = 'RIGHT'

# Moving the snake

if direction == 'UP':

snake\_position[1] -= 10

if direction == 'DOWN':

snake\_position[1] += 10

if direction == 'LEFT':

snake\_position[0] -= 10

if direction == 'RIGHT':

snake\_position[0] += 10

# Snake body growing mechanism

# if fruits and snakes collide then scores will be

# incremented by 10

snake\_body.insert(0, list(snake\_position))

if snake\_position[0] == fruit\_position[0] and snake\_position[1] == fruit\_position[1]:

score += 10

fruit\_spawn = False

else:

snake\_body.pop()

if not fruit\_spawn:

fruit\_position = [random.randrange(1, (window\_x//10)) \* 10,

random.randrange(1, (window\_y//10)) \* 10]

fruit\_spawn = True

game\_window.fill(black)

for pos in snake\_body:

pygame.draw.rect(game\_window, green, pygame.Rect(

pos[0], pos[1], 10, 10))

pygame.draw.rect(game\_window, white, pygame.Rect(

fruit\_position[0], fruit\_position[1], 10, 10))

# Game Over conditions

if snake\_position[0] < 0 or snake\_position[0] > window\_x-10:

game\_over()

if snake\_position[1] < 0 or snake\_position[1] > window\_y-10:

game\_over()

# Touching the snake body

for block in snake\_body[1:]:

if snake\_position[0] == block[0] and snake\_position[1] == block[1]:

game\_over()

# displaying score continuously

show\_score(1, white, 'times new roman', 20)

# Refresh game screen

pygame.display.update()

# Frame Per Second /Refresh Rate

fps.tick(snake\_speed)

Below is the Implementation

# importing libraries

import pygame

import time

import random

snake\_speed = 15

# Window size

window\_x = 720

window\_y = 480

# defining colors

black = pygame.Color(0, 0, 0)

white = pygame.Color(255, 255, 255)

red = pygame.Color(255, 0, 0)

green = pygame.Color(0, 255, 0)

blue = pygame.Color(0, 0, 255)

# Initialising pygame

pygame.init()

# Initialise game window

pygame.display.set\_caption('GeeksforGeeks Snakes')

game\_window = pygame.display.set\_mode((window\_x, window\_y))

# FPS (frames per second) controller

fps = pygame.time.Clock()

# defining snake default position

snake\_position = [100, 50]

# defining first 4 blocks of snake body

snake\_body = [[100, 50],

[90, 50],

[80, 50],

[70, 50]

]

# fruit position

fruit\_position = [random.randrange(1, (window\_x//10)) \* 10,

random.randrange(1, (window\_y//10)) \* 10]

fruit\_spawn = True

# setting default snake direction towards

# right

direction = 'RIGHT'

change\_to = direction

# initial score

score = 0

# displaying Score function

def show\_score(choice, color, font, size):

# creating font object score\_font

score\_font = pygame.font.SysFont(font, size)

# create the display surface object

# score\_surface

score\_surface = score\_font.render('Score : ' + str(score), True, color)

# create a rectangular object for the text

# surface object

score\_rect = score\_surface.get\_rect()

# displaying text

game\_window.blit(score\_surface, score\_rect)

# game over function

def game\_over():

# creating font object my\_font

my\_font = pygame.font.SysFont('times new roman', 50)

# creating a text surface on which text

# will be drawn

game\_over\_surface = my\_font.render(

'Your Score is : ' + str(score), True, red)

# create a rectangular object for the text

# surface object

game\_over\_rect = game\_over\_surface.get\_rect()

# setting position of the text

game\_over\_rect.midtop = (window\_x/2, window\_y/4)

# blit will draw the text on screen

game\_window.blit(game\_over\_surface, game\_over\_rect)

pygame.display.flip()

# after 2 seconds we will quit the program

time.sleep(2)

# deactivating pygame library

pygame.quit()

# quit the program

quit()

# Main Function

while True:

# handling key events

for event in pygame.event.get():

if event.type == pygame.KEYDOWN:

if event.key == pygame.K\_UP:

change\_to = 'UP'

if event.key == pygame.K\_DOWN:

change\_to = 'DOWN'

if event.key == pygame.K\_LEFT:

change\_to = 'LEFT'

if event.key == pygame.K\_RIGHT:

change\_to = 'RIGHT'

# If two keys pressed simultaneously

# we don't want snake to move into two

# directions simultaneously

if change\_to == 'UP' and direction != 'DOWN':

direction = 'UP'

if change\_to == 'DOWN' and direction != 'UP':

direction = 'DOWN'

if change\_to == 'LEFT' and direction != 'RIGHT':

direction = 'LEFT'

if change\_to == 'RIGHT' and direction != 'LEFT':

direction = 'RIGHT'

# Moving the snake

if direction == 'UP':

snake\_position[1] -= 10

if direction == 'DOWN':

snake\_position[1] += 10

if direction == 'LEFT':

snake\_position[0] -= 10

if direction == 'RIGHT':

snake\_position[0] += 10

# Snake body growing mechanism

# if fruits and snakes collide then scores

# will be incremented by 10

snake\_body.insert(0, list(snake\_position))

if snake\_position[0] == fruit\_position[0] and snake\_position[1] == fruit\_position[1]:

score += 10

fruit\_spawn = False

else:

snake\_body.pop()

if not fruit\_spawn:

fruit\_position = [random.randrange(1, (window\_x//10)) \* 10,

random.randrange(1, (window\_y//10)) \* 10]

fruit\_spawn = True

game\_window.fill(black)

for pos in snake\_body:

pygame.draw.rect(game\_window, green,

pygame.Rect(pos[0], pos[1], 10, 10))

pygame.draw.rect(game\_window, white, pygame.Rect(

fruit\_position[0], fruit\_position[1], 10, 10))

# Game Over conditions

if snake\_position[0] < 0 or snake\_position[0] > window\_x-10:

game\_over()

if snake\_position[1] < 0 or snake\_position[1] > window\_y-10:

game\_over()

# Touching the snake body

for block in snake\_body[1:]:

if snake\_position[0] == block[0] and snake\_position[1] == block[1]:

game\_over()

# displaying score continuously

show\_score(1, white, 'times new roman', 20)

# Refresh game screen

pygame.display.update()

# Frame Per Second /Refresh Rate

fps.tick(snake\_speed)

### Output:

A black and orange circle with white background

AI-generated content may be incorrect.

Snake using Pygame

### Code Explanation:

1. Kód začína importom potrebných knižníc.
2. Sú to pygame, čas a náhoda. (pygame, time a random)
3. Ďalej kód definuje niektoré premenné.
4. Premenná **snake\_speed** riadi, ako rýchlo sa had pohybuje po obrazovke.
5. Premenné window\_x a window\_y definujú veľkosť herného okna na obrazovke.
6. Nasledujúci riadok kódu inicializuje pygame.
7. Je to dôležité, pretože nastavuje všetky herné objekty a ich vlastnosti tak, aby ich bolo možné neskôr použiť v programe.
8. Ďalej kód vytvorí inštanciu objektu triedy Game\_Window pygame.
9. Tento objekt predstavuje obdĺžnikovú oblasť na obrazovke, ktorú je možné vyplniť grafikou a textovým obsahom.
10. Objekt Game\_Window má dve vlastnosti: šírku a výšku.
11. Tieto hodnoty znázorňujú, aké široké a vysoké je herné okno.
12. Nasledujúci riadok kódu priraďuje hodnoty týmto vlastnostiam na základe hodnoty definovanej používateľom s názvom snake\_speed.
13. Táto variabilná hovorí, ako rýchlo (v pixeloch za sekundu) pohybovať hadom po obrazovke.
14. Vyššie hodnoty umožnia rýchlejší pohyb, ale aj intenzívnejšiu hrateľnosť!
15. Ďalej PyGame začne načítavať rôzne obrázky do pamäte, ktoré sa použijú ako grafika na pozadí pre náš herný svet.
16. Najprv sa načíta v
17. Kód vytvorí okno s rozmermi 720x480 pixelov.
18. Farby čierna, biela, červená, zelená a modrá budú použité na znázornenie rôznych prvkov hry.
19. Ďalej sa importuje a inicializuje modul pygame.
20. To nám umožní začať pracovať s rôznymi objektmi a funkciami hry.
21. Hlavná slučka hry sa potom spustí volaním pygame.init().
22. Táto funkcia zabezpečí, že všetky potrebné moduly sú načítané a pripravené na použitie.
23. Nakoniec zavoláme konštruktor okna, aby vytvoril naše herné okno.
24. Kód začína vytvorením funkcie pygame.display.set\_mode() na nastavenie veľkosti a polohy okna.
25. Kód potom vytvorí herné okno a nastaví jeho režim na (0, 0).
26. Ďalej kód definuje niektoré premenné: fps, snake\_position, snake\_body a fruit\_position.
27. Tieto premenné sa použijú na kontrolu rýchlosti hada, odkiaľ začína (snake\_position), aká je široká (snake\_body), kde sa nachádza ovocie (fruit\_position) a či by sa ovocie malo alebo nemalo rozmnožovať (fruit\_spawn).
28. Ďalší blok kódu vypočíta vzdialenosť medzi jednotlivými bodmi na obrazovke pomocou pygame.time.Clock().
29. To nám umožňuje pohybovať hadom na obrazovke bez toho, aby sme museli neustále prepočítavať jeho polohu.
30. Nakoniec nastavíme dve booleovské premenné: fruit\_spawn a analyze().
31. Tie určia, či sa ovocie objaví na náhodných miestach na obrazovke a bude analyzované pre vstup hráča.
32. Kód nastavuje základné herné okno s hadom umiestneným na (100, 50) na osi X a (window\_x, window\_y) na osi Y.
33. Ovládač FPS je inicializovaný a nastavený tak, aby bežal rýchlosťou 60 snímok za sekundu.
34. Ďalší blok kódu definuje telo hada.
35. Vytvorí sa zoznam desiatich [100, 50] bodov, počnúc pozíciou (100, 50).
36. Prvé štyri body sú nastavené tak, aby boli v strede hadieho tela, zatiaľ čo zvyšných šesť bodov je rovnomerne rozmiestnených okolo neho.
37. Ďalej je pozícia ovocia definovaná ako [(random.randrange(1, (window\_x//10)) \* 10), (random.randrange(1
38. Kód začína inicializáciou niektorých premenných.
39. Prvým je skóre, ktoré začína na 0.
40. Druhým je smerová premenná, ktorá určí, ako sa had bude pohybovať.
41. Funkcia show\_score() sa volá vždy, keď hráč urobí voľbu.
42. Táto funkcia sa skladá z troch častí: vytvorenie objektu písma, vytvorenie objektu zobrazovanej plochy a zobrazenie textu na zobrazovacej ploche.
43. Najprv sa vytvorí score\_font objekt.
44. Tento objekt ukladá informácie o písme použitom na zobrazenie textu na obrazovke (v tomto prípade Times New Roman).
45. Ďalej sa vytvorí score\_surface objekt a inicializuje sa s informáciami o písme a veľkosti textu, ktorý sa zobrazí (veľkosť 50 bodov).
46. Nakoniec sa pomocou blit() objekt score\_rect skopíruje na score\_surface objekt, aby sa mohol zobraziť na obrazovke.
47. Funkcia game\_over() ukončí akúkoľvek aktuálnu hru a ukončí kód Pythonu bežiaci v tomto module (za predpokladu, že ho nevolajú žiadne iné funkcie).
48. Najprv sa vytvorí inštancia písma SysFont s názvom my\_font.
49. Potom sa zadá veľkosť 50 bodov pre Times New Roman ako jeho typ písma a farebné hodnoty.
50. Nakoniec sa zavolá game over(), aby ukončil všetky hry a ukončil kód Pythonu spustený v
51. Kód najprv inicializuje niektoré premenné vrátane premennej skóre.
52. Kód potom vytvorí funkciu s názvom show\_score().
53. Táto funkcia sa použije na zobrazenie aktuálneho skóre na obrazovke.
54. Funkcia show\_score() najprv vytvorí objekt písma s názvom score\_font a nastaví jeho veľkosť na 50 bodov.
55. Ďalej funkcia vytvorí objekt zobrazovanej plochy s názvom score\_surface a nastaví jeho farbu na bielu.
56. Nakoniec funkcia show\_score() vyfúkne score\_surface objekt na obrazovku okna hry.
57. Funkcia game over() je zodpovedná za čistenie zdrojov po skončení hry.
58. Najprv vytvorí objekt písma s názvom my\_font a nastaví jeho veľkosť na 20 bodov.
59. Potom sa zobrazí funkcia game over() bl
60. Kód najprv vytvorí objekt textovej plochy s názvom game\_over\_surface.
61. Text sa vykreslí v písme my\_font a červenou farbou.
62. Ďalej sa vytvorí obdĺžnikový objekt pre objekt textového povrchu.
63. Tento objekt bude mať svoj stred v (window\_x/2, window\_y/4).
64. Nakoniec sa poloha textu na obdĺžniku nastaví pomocou game\_over\_rect.midtop().
65. Kód vytvorí objekt povrchovej vrstvy textu s názvom game\_over\_surface.
66. Tento objekt sa použije na zobrazenie skóre hráča a správy "Vaše skóre je :".
67. Ďalej sa vytvorí obdĺžnikový objekt s názvom game\_over\_rect.
68. Tento objekt sa použije na umiestnenie textu na povrchu.
69. Stred obdĺžnika je nastavený na (window\_x/2, window\_y/4).
70. Kód začína inicializáciou knižnice pygame.
71. Ďalej kód vytvorí okno a priradí ho k game\_window.
72. Okno má povrch (grafické znázornenie obrazovky) a objekt Obdĺžnik, ktorý určuje jeho veľkosť a polohu.
73. Ďalej kód prenesie text "GAME OVER" na game\_over\_surface objekt.
74. Text je nakreslený bielou farbou v strede nad game\_over\_rect objektom.
75. Program potom nastaví časovač, ktorý pobeží 2 sekundy.
76. V tomto bode sa program ukončí, pretože už nie je potrebné spustiť žiadny kód.
77. Kód skontroluje kľúčové udalosti a ak udalosť zodpovedá platnému kľúču, zodpovedajúcim spôsobom zmení text zobrazený na obrazovke.
78. Ak stlačíte akékoľvek iné tlačidlo, program bude pokračovať ako zvyčajne.
79. Kód začína kontrolou, či hráč stlačil dve klávesy súčasne.
80. Ak áno, kód zmení smer hada.
81. Ďalej kód skontroluje, či niektorý z klávesov nebol stlačený iným smerom, ako sa očakávalo.
82. Ak áno, potom kód zodpovedajúcim spôsobom upraví polohu hada.
83. Nakoniec aktualizuje, aké veľké je hadie telo.
84. Kód skontroluje, či sú dve súčasne stlačené klávesy "HORE" alebo "DOLE".
85. Ak sú, smer hada sa zodpovedajúcim spôsobom zmení.
86. Ak dve stlačené klávesy nie sú rovnaké, kód skontroluje, či sú v rôznych smeroch.
87. Ak nie sú, poloha hada sa upraví o 10 pixelov v každom smere.
88. Nakoniec sa vytvorí funkcia, ktorá zmení veľkosť hadieho tela pri pohybe.
89. Kód začína vytvorením zoznamu pozícií hadov.
90. Prvá pozícia v zozname je (0, 0) a posledná pozícia v zozname je (window\_x-10, window\_y-10).
91. Ďalej kód skontroluje, či sa niektorá z pozícií v hadovi rovná polohe ovocia.
92. Ak áno, potom toto ovocie získa 10 bodov a pridá sa k premennej ovocného spawnu.
93. Ak sa nenájdu žiadne ovocie, hra sa presunie na kontrolu kolízií medzi hadmi a ovocím.
94. Ak sa dva hady pretínajú, ich skóre sa zvýši o 10.
95. Ak sa had zrazí so stenou alebo iným hadom, potom tento had zomrie a spustia sa podmienky prehry.
96. Nakoniec, dotyk ktorejkoľvek časti hada spôsobí jeho smrť a tiež spustí podmienky hry.
97. Kód skontroluje, či sú dve polohy v hadom tele rovnaké.
98. Ak áno, skóre sa zvýši o 10 a zavolá sa funkcia game\_over().
99. Ak sa hráč v ktoromkoľvek bode dotkne hadieho tela, vyvolá sa funkcia game\_over().