Table of contents

Step 1: Create empty window

- Create an empty window.
- Create main game loop.
- Implement exit action.

```
#### 1 - Create empty window #####

import pygame
from pygame.locals import *

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')
while True:
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            exit()
    pygame.display.update()
```

Step 2: Add static snake

- Create snake
- Create snake skin
- Clear screen
- Draw each coordinate

```
import pygame
from pygame.locals import *

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')

snake = [(200, 200), (210, 200), (220,200)]
snake_skin = pygame.Surface((10,10))
snake_skin.fill((255,255,255)) #White

while True:
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            exit()
    screen.fill((0,0,0))
    for pos in snake:
        screen.blit(snake_skin,pos)
    pygame.display.update()
```

Step 3: Make snake move

- At this point the snake will be moving too fast, it won't be possible to see on the screen.
- Need next step to see the snake moving.

```
import pygame
from pygame.locals import *

# Macro definition for snake movement.
UP = 0
RIGHT = 1
DOWN = 2
LEFT = 3

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')

snake = [(200, 200), (210, 200), (220,200)]
snake_skin = pygame.Surface((10,10))
snake_skin.fill((255,255,255)) #White

my_direction = LEFT

while True:
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            exit()
        if event.type == KEYDOWN:
            if event.key == K_UP:
                my_direction = UP
            if event.key == K_DOWN:
                my_direction = DOWN
            if event.key == K_LEFT:
                my_direction = LEFT
            if event.key == K_RIGHT:
                my_direction = RIGHT
```

```
    for i in range(len(snake) - 1, 0, -1):
        snake[i] = (snake[i-1][0], snake[i-1][1])

    # Actually make the snake move.
    if my_direction == UP:
        snake[0] = (snake[0][0], snake[0][1] - 10)
    if my_direction == DOWN:
        snake[0] = (snake[0][0], snake[0][1] + 10)
    if my_direction == RIGHT:
        snake[0] = (snake[0][0] + 10, snake[0][1])
    if my_direction == LEFT:
        snake[0] = (snake[0][0] - 10, snake[0][1])

    screen.fill((0,0,0))
    for pos in snake:
        screen.blit(snake_skin,pos)

    pygame.display.update()
```

Step 4: Add clock

- Make the snake move in a slower speed.
- The clock could be adjusted in the future accordingly to the snake size.

```
#### 4 - Add clock ####
import pygame
from pygame.locals import *

# Macro definition for snake movement.
UP = 0
RIGHT = 1
DOWN = 2
LEFT = 3

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')

snake = [(200, 200), (210, 200), (220,200)]
snake_skin = pygame.Surface((10,10))
snake_skin.fill((255,255,255)) #White

my_direction = LEFT

clock = pygame.time.Clock()

while True:
    clock.tick(10)
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            exit()

        if event.type == KEYDOWN:
            if event.key == K_UP:
                my_direction = UP
            if event.key == K_DOWN:
                my_direction = DOWN
            if event.key == K_LEFT:
                my_direction = LEFT
```

```
            if event.key == K_RIGHT:
                my_direction = RIGHT

    for i in range(len(snake) - 1, 0, -1):
        snake[i] = (snake[i-1][0], snake[i-1][1])

    # Actually make the snake move.
    if my_direction == UP:
        snake[0] = (snake[0][0], snake[0][1] - 10)
    if my_direction == DOWN:
        snake[0] = (snake[0][0], snake[0][1] + 10)
    if my_direction == RIGHT:
        snake[0] = (snake[0][0] + 10, snake[0][1])
    if my_direction == LEFT:
        snake[0] = (snake[0][0] - 10, snake[0][1])

    screen.fill((0,0,0))
    for pos in snake:
        screen.blit(snake_skin,pos)

    pygame.display.update()
```

Step 5: Add apple

```
#### 5 - Add apple ####
import pygame
############################NEW CODE#################################
import random
############################NEW CODE#################################
from pygame.locals import *

############################NEW CODE#################################
# Helper functions
def on_grid_random():
    x = random.randint(0,59)
    y = random.randint(0,59)
    return (x * 10, y * 10)
############################NEW CODE#################################
```

```
# Macro definition for snake movement.
UP = 0
RIGHT = 1
DOWN = 2
LEFT = 3

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')

snake = [(200, 200), (210, 200), (220,200)]
snake_skin = pygame.Surface((10,10))
snake_skin.fill((255,255,255)) #White

apple_pos = on_grid_random()
apple = pygame.Surface((10,10))
apple.fill((255,0,0))

my_direction = LEFT

clock = pygame.time.Clock()

while True:
    clock.tick(10)
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            exit()

        if event.type == KEYDOWN:
            if event.key == K_UP:
                my_direction = UP
            if event.key == K_DOWN:
                my_direction = DOWN
            if event.key == K_LEFT:
                my_direction = LEFT
            if event.key == K_RIGHT:
                my_direction = RIGHT

    for i in range(len(snake) - 1, 0, -1):
        snake[i] = (snake[i-1][0], snake[i-1][1])
```

```
    # Actually make the snake move.
    if my_direction == UP:
        snake[0] = (snake[0][0], snake[0][1] - 10)
    if my_direction == DOWN:
        snake[0] = (snake[0][0], snake[0][1] + 10)
    if my_direction == RIGHT:
        snake[0] = (snake[0][0] + 10, snake[0][1])
    if my_direction == LEFT:
        snake[0] = (snake[0][0] - 10, snake[0][1])

    screen.fill((0,0,0))
    screen.blit(apple, apple_pos)
    for pos in snake:
        screen.blit(snake_skin,pos)

    pygame.display.update()
```

Step 6: Eat apple

- Make snake grow one block every time it "eats" an apple.
- Replot the apple in a new random position.

```
#### 6 - Make snake eat apple and grow ####
import pygame, random
from pygame.locals import *

# Helper functions
def on_grid_random():
    x = random.randint(0,59)
    y = random.randint(0,59)
    return (x * 10, y * 10)

def collision(c1, c2):
    return (c1[0] == c2[0]) and (c1[1] == c2[1])

# Macro definition for snake movement.
UP = 0
RIGHT = 1
DOWN = 2
LEFT = 3

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')

snake = [(200, 200), (210, 200), (220,200)]
snake_skin = pygame.Surface((10,10))
snake_skin.fill((255,255,255)) #White

apple_pos = on_grid_random()
apple = pygame.Surface((10,10))
apple.fill((255,0,0))

my_direction = LEFT

clock = pygame.time.Clock()

while True:
```

```
    clock.tick(10)
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            exit()

        if event.type == KEYDOWN:
            if event.key == K_UP:
                my_direction = UP
            if event.key == K_DOWN:
                my_direction = DOWN
            if event.key == K_LEFT:
                my_direction = LEFT
            if event.key == K_RIGHT:
                my_direction = RIGHT

    if collision(snake[0], apple_pos):
        apple_pos = on_grid_random()
        snake.append((0,0))

    for i in range(len(snake) - 1, 0, -1):
        snake[i] = (snake[i-1][0], snake[i-1][1])

    # Actually make the snake move.
    if my_direction == UP:
        snake[0] = (snake[0][0], snake[0][1] - 10)
    if my_direction == DOWN:
        snake[0] = (snake[0][0], snake[0][1] + 10)
    if my_direction == RIGHT:
        snake[0] = (snake[0][0] + 10, snake[0][1])
    if my_direction == LEFT:
        snake[0] = (snake[0][0] - 10, snake[0][1])

    screen.fill((0,0,0))
    screen.blit(apple, apple_pos)
    for pos in snake:
        screen.blit(snake_skin,pos)

    pygame.display.update()
```

Step 7: Add collision detection

- Game ends if snake collides with edges.
- Game ends if snake collides with itself.
- Snake head is position 0 since it started going to the left.

```
##### 7 - Collision detection ####
import pygame, random
from pygame.locals import *

# Helper functions
def on_grid_random():
    x = random.randint(0,59)
    y = random.randint(0,59)
    return (x * 10, y * 10)

def collision(c1, c2):
    return (c1[0] == c2[0]) and (c1[1] == c2[1])

# Macro definition for snake movement.
UP = 0
RIGHT = 1
DOWN = 2
LEFT = 3

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')

snake = [(200, 200), (210, 200), (220,200)]
snake_skin = pygame.Surface((10,10))
snake_skin.fill((255,255,255)) #White

apple_pos = on_grid_random()
apple = pygame.Surface((10,10))
apple.fill((255,0,0))

my_direction = LEFT

clock = pygame.time.Clock()
```

```
while True:
    clock.tick(10)
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            exit()

        if event.type == KEYDOWN:
            if event.key == K_UP and my_direction != DOWN:
                my_direction = UP
            if event.key == K_DOWN and my_direction != UP:
                my_direction = DOWN
            if event.key == K_LEFT and my_direction != RIGHT:
                my_direction = LEFT
            if event.key == K_RIGHT and my_direction != LEFT:
                my_direction = RIGHT


    if collision(snake[0], apple_pos):
        apple_pos = on_grid_random()
        snake.append((0,0))

    # Check if snake collided with boundaries
    if snake[0][0] == 600 or snake[0][1] == 600 or snake[0][0] < 0 or
snake[0][1] < 0:
        pygame.quit()
        exit()
    # Check if the snake has hit itself
    for i in range(1, len(snake) - 1):
        if snake[0][0] == snake[i][0] and snake[0][1] == snake[i][1]:
            pygame.quit()
            exit()

    for i in range(len(snake) - 1, 0, -1):
        snake[i] = (snake[i-1][0], snake[i-1][1])

    # Actually make the snake move.
    if my_direction == UP:
        snake[0] = (snake[0][0], snake[0][1] - 10)
    if my_direction == DOWN:
        snake[0] = (snake[0][0], snake[0][1] + 10)
    if my_direction == RIGHT:
```

```
        snake[0] = (snake[0][0] + 10, snake[0][1])
    if my_direction == LEFT:
        snake[0] = (snake[0][0] - 10, snake[0][1])

    screen.fill((0,0,0))
    screen.blit(apple, apple_pos)
    for pos in snake:
        screen.blit(snake_skin,pos)

    pygame.display.update()
```

Step 8: Add grid

```
##### 8 - Add grid ####
import pygame, random
from pygame.locals import *

# Helper functions
def on_grid_random():
    x = random.randint(0,59)
    y = random.randint(0,59)
    return (x * 10, y * 10)

def collision(c1, c2):
    return (c1[0] == c2[0]) and (c1[1] == c2[1])

# Macro definition for snake movement.
UP = 0
RIGHT = 1
DOWN = 2
LEFT = 3

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')

snake = [(200, 200), (210, 200), (220,200)]
snake_skin = pygame.Surface((10,10))
snake_skin.fill((255,255,255)) #White

apple_pos = on_grid_random()
apple = pygame.Surface((10,10))
apple.fill((255,0,0))

my_direction = LEFT

clock = pygame.time.Clock()

while True:
    clock.tick(10)
    for event in pygame.event.get():
        if event.type == QUIT:
```

```
            pygame.quit()
            exit()

        if event.type == KEYDOWN:
            if event.key == K_UP and my_direction != DOWN:
                my_direction = UP
            if event.key == K_DOWN and my_direction != UP:
                my_direction = DOWN
            if event.key == K_LEFT and my_direction != RIGHT:
                my_direction = LEFT
            if event.key == K_RIGHT and my_direction != LEFT:
                my_direction = RIGHT

    if collision(snake[0], apple_pos):
        apple_pos = on_grid_random()
        snake.append((0,0))

   # Check if snake collided with boundaries
    if snake[0][0] == 600 or snake[0][1] == 600 or snake[0][0] < 0 or
snake[0][1] < 0:
        pygame.quit()
        exit()
    # Check if the snake has hit itself
    for i in range(1, len(snake) - 1):
        if snake[0][0] == snake[i][0] and snake[0][1] == snake[i][1]:
            pygame.quit()
            exit()

    for i in range(len(snake) - 1, 0, -1):
        snake[i] = (snake[i-1][0], snake[i-1][1])

    # Actually make the snake move.
    if my_direction == UP:
        snake[0] = (snake[0][0], snake[0][1] - 10)
    if my_direction == DOWN:
        snake[0] = (snake[0][0], snake[0][1] + 10)
    if my_direction == RIGHT:
        snake[0] = (snake[0][0] + 10, snake[0][1])
    if my_direction == LEFT:
        snake[0] = (snake[0][0] - 10, snake[0][1])

    screen.fill((0,0,0))
```

```
    screen.blit(apple, apple_pos)

    for x in range(0, 600, 10): # Draw vertical lines
        pygame.draw.line(screen, (40, 40, 40), (x, 0), (x, 600))
    for y in range(0, 600, 10): # Draw vertical lines
        pygame.draw.line(screen, (40, 40, 40), (0, y), (600, y))

    for pos in snake:
        screen.blit(snake_skin,pos)

    pygame.display.update()
```

Step 9: Add score

```
##### 9 - Add score ####
import pygame, random
from pygame.locals import *

# Helper functions
def on_grid_random():
    x = random.randint(0,59)
    y = random.randint(0,59)
    return (x * 10, y * 10)

def collision(c1, c2):
    return (c1[0] == c2[0]) and (c1[1] == c2[1])

# Macro definition for snake movement.
UP = 0
RIGHT = 1
DOWN = 2
LEFT = 3

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')

snake = [(200, 200), (210, 200), (220,200)]
snake_skin = pygame.Surface((10,10))
snake_skin.fill((255,255,255)) #White

apple_pos = on_grid_random()
apple = pygame.Surface((10,10))
apple.fill((255,0,0))

my_direction = LEFT

clock = pygame.time.Clock()

font = pygame.font.Font('freesansbold.ttf', 18)
score = 0

while True:
```

```
    clock.tick(10)
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            exit()

        if event.type == KEYDOWN:
            if event.key == K_UP and my_direction != DOWN:
                my_direction = UP
            if event.key == K_DOWN and my_direction != UP:
                my_direction = DOWN
            if event.key == K_LEFT and my_direction != RIGHT:
                my_direction = LEFT
            if event.key == K_RIGHT and my_direction != LEFT:
                my_direction = RIGHT

    if collision(snake[0], apple_pos):
        apple_pos = on_grid_random()
        snake.append((0,0))
        score = score + 1

   # Check if snake collided with boundaries
    if snake[0][0] == 600 or snake[0][1] == 600 or snake[0][0] < 0 or
snake[0][1] < 0:
        pygame.quit()
        exit()
    # Check if the snake has hit itself
    for i in range(1, len(snake) - 1):
        if snake[0][0] == snake[i][0] and snake[0][1] == snake[i][1]:
            pygame.quit()
            exit()

    for i in range(len(snake) - 1, 0, -1):
        snake[i] = (snake[i-1][0], snake[i-1][1])

    # Actually make the snake move.
    if my_direction == UP:
        snake[0] = (snake[0][0], snake[0][1] - 10)
    if my_direction == DOWN:
        snake[0] = (snake[0][0], snake[0][1] + 10)
    if my_direction == RIGHT:
        snake[0] = (snake[0][0] + 10, snake[0][1])
```

```
    if my_direction == LEFT:
        snake[0] = (snake[0][0] - 10, snake[0][1])

    screen.fill((0,0,0))
    screen.blit(apple, apple_pos)

    for x in range(0, 600, 10): # Draw vertical lines
        pygame.draw.line(screen, (40, 40, 40), (x, 0), (x, 600))
    for y in range(0, 600, 10): # Draw vertical lines
        pygame.draw.line(screen, (40, 40, 40), (0, y), (600, y))

    score_font = font.render('Score: %s' % (score), True, (255, 255, 255))
    score_rect = score_font.get_rect()
    score_rect.topleft = (600 - 120, 10)
    screen.blit(score_font, score_rect)

    for pos in snake:
        screen.blit(snake_skin,pos)

    pygame.display.update()
```

Step 10: Add game over

```
##### 10 - Game over ####
import pygame, random
from pygame.locals import *

# Helper functions
def on_grid_random():
    x = random.randint(0,59)
    y = random.randint(0,59)
    return (x * 10, y * 10)

def collision(c1, c2):
    return (c1[0] == c2[0]) and (c1[1] == c2[1])

# Macro definition for snake movement.
UP = 0
RIGHT = 1
DOWN = 2
LEFT = 3

pygame.init()
screen = pygame.display.set_mode((600, 600))
pygame.display.set_caption('Snake')

snake = [(200, 200), (210, 200), (220,200)]
snake_skin = pygame.Surface((10,10))
snake_skin.fill((255,255,255)) #White

apple_pos = on_grid_random()
apple = pygame.Surface((10,10))
apple.fill((255,0,0))

my_direction = LEFT

clock = pygame.time.Clock()

font = pygame.font.Font('freesansbold.ttf', 18)
score = 0
```

```
game_over = False
while not game_over:
    clock.tick(10)
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            exit()

        if event.type == KEYDOWN:
            if event.key == K_UP and my_direction != DOWN:
                my_direction = UP
            if event.key == K_DOWN and my_direction != UP:
                my_direction = DOWN
            if event.key == K_LEFT and my_direction != RIGHT:
                my_direction = LEFT
            if event.key == K_RIGHT and my_direction != LEFT:
                my_direction = RIGHT

    if collision(snake[0], apple_pos):
        apple_pos = on_grid_random()
        snake.append((0,0))
        score = score + 1

    # Check if snake collided with boundaries
    if snake[0][0] == 600 or snake[0][1] == 600 or snake[0][0] < 0 or
snake[0][1] < 0:
        game_over = True
        break

    # Check if the snake has hit itself
    for i in range(1, len(snake) - 1):
        if snake[0][0] == snake[i][0] and snake[0][1] == snake[i][1]:
            game_over = True
            break

    if game_over:
        break

    for i in range(len(snake) - 1, 0, -1):
        snake[i] = (snake[i-1][0], snake[i-1][1])

    # Actually make the snake move.
```

```
    if my_direction == UP:
        snake[0] = (snake[0][0], snake[0][1] - 10)
    if my_direction == DOWN:
        snake[0] = (snake[0][0], snake[0][1] + 10)
    if my_direction == RIGHT:
        snake[0] = (snake[0][0] + 10, snake[0][1])
    if my_direction == LEFT:
        snake[0] = (snake[0][0] - 10, snake[0][1])

    screen.fill((0,0,0))
    screen.blit(apple, apple_pos)

    for x in range(0, 600, 10): # Draw vertical lines
        pygame.draw.line(screen, (40, 40, 40), (x, 0), (x, 600))
    for y in range(0, 600, 10): # Draw vertical lines
        pygame.draw.line(screen, (40, 40, 40), (0, y), (600, y))

    score_font = font.render('Score: %s' % (score), True, (255, 255, 255))
    score_rect = score_font.get_rect()
    score_rect.topleft = (600 - 120, 10)
    screen.blit(score_font, score_rect)

    for pos in snake:
        screen.blit(snake_skin,pos)

    pygame.display.update()

while True:
    game_over_font = pygame.font.Font('freesansbold.ttf', 75)
    game_over_screen = game_over_font.render('Game Over', True, (255, 255,
255))
    game_over_rect = game_over_screen.get_rect()
    game_over_rect.midtop = (600 / 2, 10)
    screen.blit(game_over_screen, game_over_rect)
    pygame.display.update()
    pygame.time.wait(500)
    while True:
        for event in pygame.event.get():
            if event.type == QUIT:
                pygame.quit()
                exit()
```