SNAKE-GAME

```
import customtkinter as ctk
from settings import *
from random import randint
from sys import exit

class Game(ctk.CTk):
    def __init__(self):
        # setup
        super().__init__()
        self.title('Snake')
        self.geometry(f'{WINDOW_SIZE[0]}x{WINDOW_SIZE[1]}')

        # layout
        self.columnconfigure(list(range(FIELDS[0])), weight=1, uniform='a')
        self.rowconfigure(list(range(FIELDS[1])), weight=1, uniform='a')

        # snake
        self.snake = [START_POS, (START_POS[0] - 1, START_POS[1]), (START_POS[0] - 2, START_POS[1])]
        self.direction = DIRECTIONS['right']
        self.bind("<Key>", self.move_snake)

        # add the apple to the grid
        self.place_apple()

        # draw logic
        self.draw_frames = []
        self.animate()
        # run
        self.mainloop()

    def move_snake(self, event):
        match event.keycode:
            case 113: self.direction = DIRECTIONS['left'] if self.direction != DIRECTIONS['right'] \
                else self.direction
            case 111: self.direction = DIRECTIONS['up'] if self.direction != DIRECTIONS['down'] \
                else self.direction
            case 114: self.direction = DIRECTIONS['right'] if self.direction != DIRECTIONS['left'] \
                else self.direction
            case 116: self.direction = DIRECTIONS['down'] if self.direction != DIRECTIONS['up'] \
                else self.direction
    def animate(self):
        # update the snake pos
        new_head = (self.snake[0][0] + self.direction[0], self.snake[0][1] + self.direction[1])
        self.snake.insert(0, new_head)

        # if there snake eats apple place the app into new grid and grow the snake  else don't grow the snake
        if self.snake[0] == self.apple_pos:
            self.place_apple()
        else:
            self.snake.pop()

        self.check_game_over()

        # draw the new snake
        self.draw()
        self.after(250, self.animate)
```

```
    def check_game_over(self):
        snake_head = self.snake[0]
        if (snake_head[0] >= RIGHT_LIMIT or snake_head[1] >= BOTTOM_LIMIT or
                snake_head[0] < LEFT_LIMIT or snake_head[1] < TOP_LIMIT or
                snake_head in self.snake[1:]):
            self.destroy()
            exit()

    def place_apple(self):
        self.apple_pos = (randint(0, FIELDS[0] - 1), randint(0, FIELDS[1] - 1))

    def draw(self):
        if self.draw_frames:
            for frame, _ in self.draw_frames:
                frame.grid_forget()
            self.draw_frames.clear()

        apple_frame = ctk.CTkFrame(self, fg_color=APPLE_COLOR)
        self.draw_frames.append((apple_frame, self.apple_pos))

        for index, pos in enumerate(self.snake):
            color = SNAKE_BODY_COLOR if index != 0 else SNAKE_HEAD_COLOR
            snake_frame = ctk.CTkFrame(self, fg_color=color, corner_radius=0)
            self.draw_frames.append((snake_frame, pos))

        for frame, pos in self.draw_frames:
            col, row = pos
            frame.grid(row=row, column=col)


Game()

# SETTINGS
# window info
WINDOW_SIZE = (800, 600)
FIELDS = (20, 15)

# movement
START_POS = (5, int(FIELDS[1] / 2))
DIRECTIONS = {'left': [-1, 0], 'right': [1, 0], 'up': [0, -1], 'down': [0, 1]}
REFRESH_SPEED = 250

# field limits
LEFT_LIMIT = -1
TOP_LIMIT = 0
RIGHT_LIMIT = FIELDS[0]
BOTTOM_LIMIT = FIELDS[1]

# colors
SNAKE_BODY_COLOR = '#8EF249'
SNAKE_HEAD_COLOR = '#71CC1D'
APPLE_COLOR = '#F9473E'
```