Projects Highlighted

# 1. Weight Converter

This project converts weight between kilograms and pounds. The user inputs the weight and specifies the unit (K for kilograms or L for pounds), and the program converts the weight to the other unit.

## Code Snippet:

weight=float(input("Enter your weight : "))  
unit=input("Enter kilogram or Pound(K/L)")  
  
if unit=='K':  
 weight=weight\*2.205  
 unit='Lbs'  
elif unit=='L':  
 weight=weight/2.205  
 unit='Kg'  
else:  
 print(f"{unit} is not valid")  
  
print(f"This weight is {round(weight)}{unit}")

# 2. Student Management System

This project represents a student with attributes like first name, last name, roll number, and stream. It includes methods to display the student's full name and stream.

## Code Snippet:

class Student:  
  
 type='University' #class variable  
  
 def \_\_init\_\_(self,fname,lname,roll,stream):  
 self.fname=fname #instance variable  
 self.lname=lname #instance variable  
 self.roll=roll #instance variable  
 self.stream=stream #instance variable  
  
 def dec(self):  
 print(f"{self.fname} {self.lname} is a student of {self.stream}")

# 3. Pygame Snake Game

This project is a simple implementation of the classic Snake game using the Pygame library. The player controls the snake to collect food, and the snake grows longer each time it eats food. The game ends if the snake runs into itself.

## Code Snippet:

import pygame  
import sys  
import random  
  
# Initialize Pygame  
pygame.init()  
  
# Constants  
WIDTH, HEIGHT = 600, 400  
GRID\_SIZE = 20  
FPS = 10  
  
# Colors  
WHITE = (255, 255, 255)  
RED = (255, 0, 0)  
GREEN = (0, 255, 0)  
  
# Snake class  
class Snake:  
 def \_\_init\_\_(self):  
 self.length = 1  
 self.positions = [((WIDTH // 2), (HEIGHT // 2))]  
 self.direction = random.choice([UP, DOWN, LEFT, RIGHT])  
 self.color = GREEN  
  
 def get\_head\_position(self):  
 return self.positions[0]  
  
 def update(self):  
 cur = self.get\_head\_position()  
 x, y = self.direction  
 new = (((cur[0] + (x \* GRID\_SIZE)) % WIDTH), (cur[1] + (y \* GRID\_SIZE)) % HEIGHT)  
 if len(self.positions) > 2 and new in self.positions[2:]:  
 self.reset()  
 else:  
 self.positions.insert(0, new)  
 if len(self.positions) > self.length:  
 self.positions.pop()  
  
 def reset(self):  
 self.length = 1  
 self.positions = [((WIDTH // 2), (HEIGHT // 2))]  
 self.direction = random.choice([UP, DOWN, LEFT, RIGHT])  
  
 def render(self, surface):  
 for p in self.positions:  
 pygame.draw.rect(surface, self.color, (p[0], p[1], GRID\_SIZE, GRID\_SIZE))  
  
# Food class  
class Food:  
 def \_\_init\_\_(self):  
 self.position = (0, 0)  
 self.color = RED  
 self.randomize\_position()  
  
 def randomize\_position(self):  
 self.position = (random.randint(0, (WIDTH // GRID\_SIZE) - 1) \* GRID\_SIZE,  
 random.randint(0, (HEIGHT // GRID\_SIZE) - 1) \* GRID\_SIZE)  
  
 def render(self, surface):  
 pygame.draw.rect(surface, self.color, (self.position[0], self.position[1], GRID\_SIZE, GRID\_SIZE))  
  
# Directions  
UP = (0, -1)  
DOWN = (0, 1)  
LEFT = (-1, 0)  
RIGHT = (1, 0)  
  
# Main function  
def main():  
 clock = pygame.time.Clock()  
 screen = pygame.display.set\_mode((WIDTH, HEIGHT), 0, 32)  
 surface = pygame.Surface(screen.get\_size())  
 surface = surface.convert()  
  
 snake = Snake()  
 food = Food()  
  
 while True:  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 sys.exit()  
 elif event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_UP:  
 if snake.direction != DOWN:  
 snake.direction = UP  
 elif event.key == pygame.K\_DOWN:  
 if snake.direction != UP:  
 snake.direction = DOWN  
 elif event.key == pygame.K\_LEFT:  
 if snake.direction != RIGHT:  
 snake.direction = LEFT  
 elif event.key == pygame.K\_RIGHT:  
 if snake.direction != LEFT:  
 snake.direction = RIGHT  
  
 snake.update()  
 if snake.get\_head\_position() == food.position:  
 snake.length += 1  
 food.randomize\_position()  
  
 surface.fill(WHITE)  
 snake.render(surface)  
 food.render(surface)  
 screen.blit(surface, (0, 0))  
 pygame.display.update()  
 clock.tick(FPS)  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()