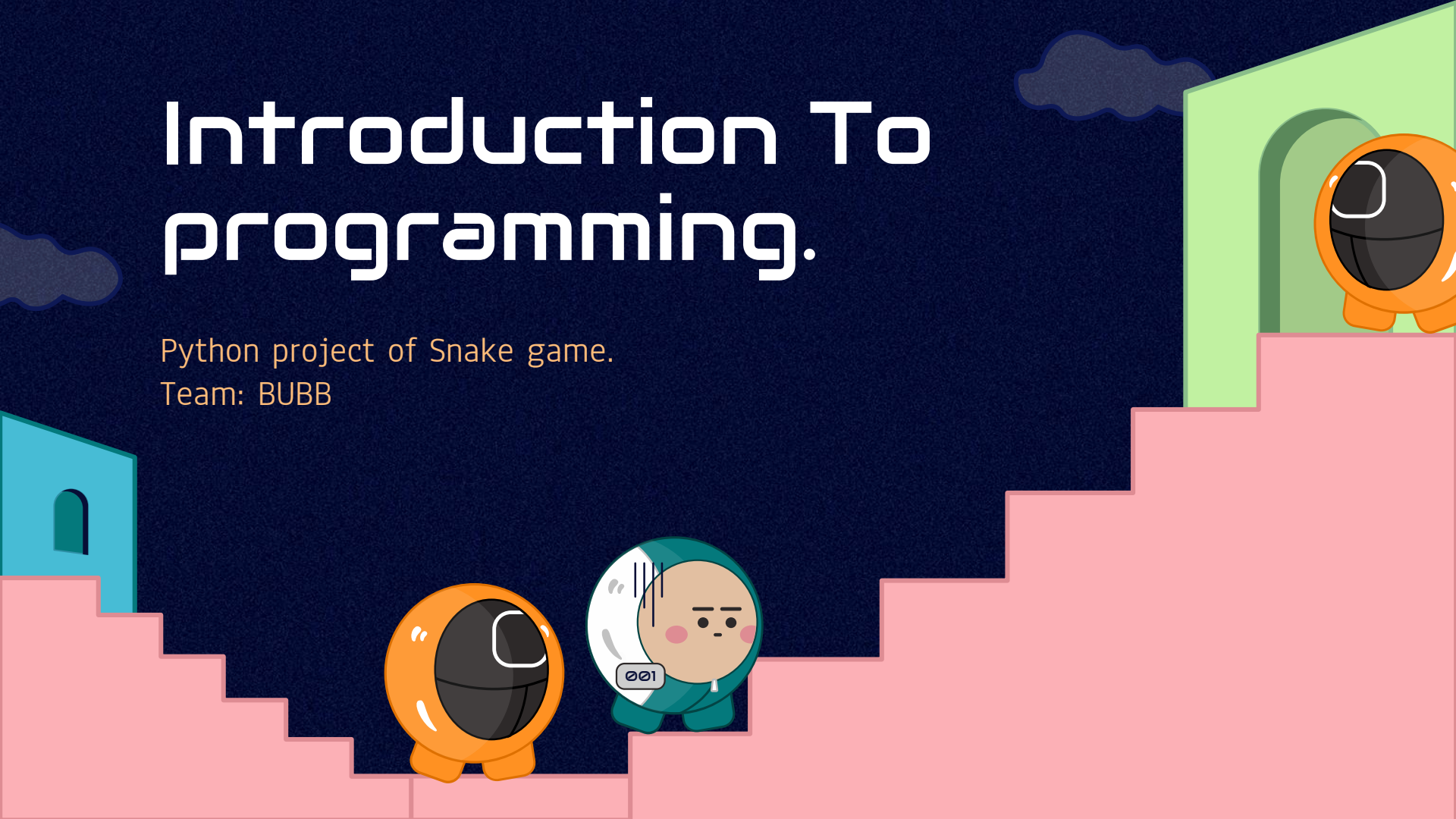
Introduction To programming.

Python project of Snake game.
Team: BUBB



THE PLAYERS

001

Mehadi Hasan Shawon-202212218

Shokrukh Shaydullo Ugli Yodgorov-202212136

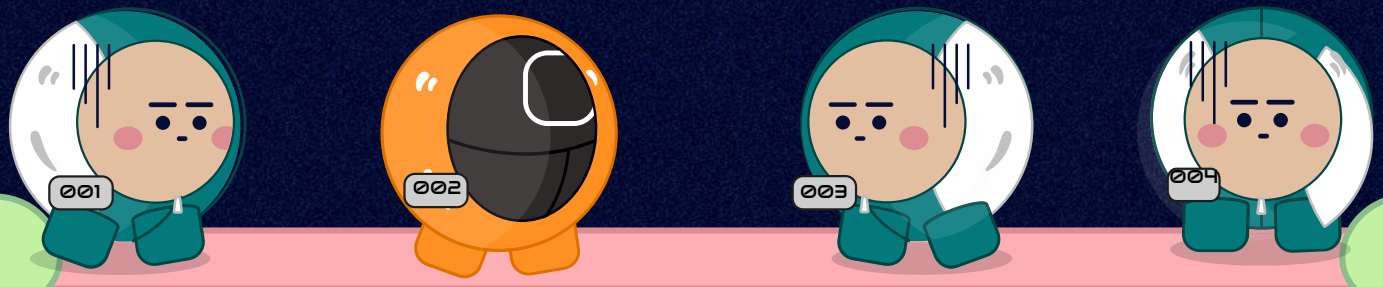
002

003

Ebrahim Khalil-202212215

Md Mujahidul Islam-202212156

004



THE GAMES STEPS

001

Game Initialization:

002

Snake Class:

003

Food Class:

004

Game Logic:

005

Control Functions:

006

Game Window and Key Bindings:

Step:01
Game Initialization:

```python
#Game Initialization:
from tkinter import *
import random

WIDTH = 500
HEIGHT = 500
SPEED = 200
SPACE_SIZE = 20
BODY_SIZE = 2
SNAKE = "#00FF00"
FOOD = "#FFFFFF"
BACKGROUND = "#000000"
```

Step:02
Snake Class:

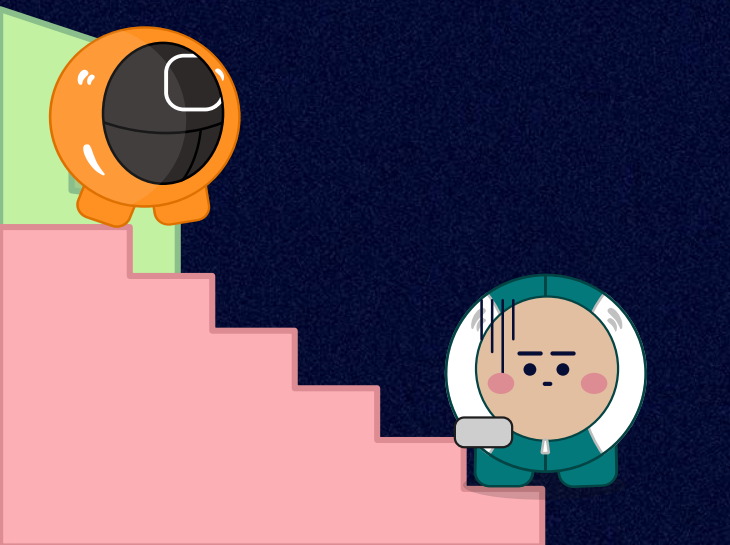
```python
#Snake Class:
class Snake:
    def __init__(self):
        # Initialize snake's properties
        self.body_size = BODY_SIZE
        self.coordinates = []
        self.squares = []

        # Create snake's initial body
        for i in range(0, BODY_SIZE):
            self.coordinates.append([0, 0])

        for x, y in self.coordinates:
            square = canvas.create_rectangle(
                x, y, x + SPACE_SIZE, y + SPACE_SIZE, fill=SNAKE, tag="snake"
            )
            self.squares.append(square)

    def increase_size(self):
        # Increase snake's size
        x, y = self.coordinates[-1]
        self.coordinates.append([x, y])
        square = canvas.create_rectangle(
            x, y, x + SPACE_SIZE, y + SPACE_SIZE, fill=SNAKE
        )
        self.squares.append(square)
```

Step:03
Food Class:

```python
#Food Class:
class Food:
    def __init__(self):
        # Generate random coordinates for the food
        x = random.randint(0, (WIDTH / SPACE_SIZE) - 1) * SPACE_SIZE
        y = random.randint(0, (HEIGHT / SPACE_SIZE) - 1) * SPACE_SIZE

        self.coordinates = [x, y]
        self.direction = random.choice(["left", "right", "up", "down"])

        canvas.create_oval(
            x, y, x + SPACE_SIZE, y + SPACE_SIZE, fill=FOOD, tag="food"
        )

    def move(self):
        # Move the food in its current direction
        x, y = self.coordinates

        if self.direction == "left":
            x -= SPACE_SIZE
        elif self.direction == "right":
            x += SPACE_SIZE
        elif self.direction == "up":
            y -= SPACE_SIZE
        elif self.direction == "down":
            y += SPACE_SIZE

        self.coordinates = [x, y]

        canvas.move(
            "food", x - canvas.coords("food")[0], y - canvas.coords("food")[1]
        )

    def change_direction(self, new_direction):
        # Change the direction of the food
        if new_direction in ["left", "right", "up", "down"]:
            self.direction = new_direction
```

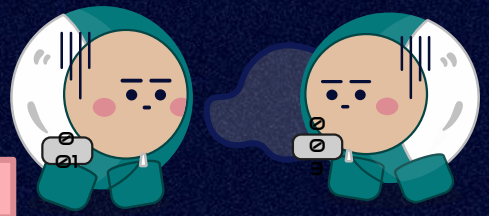
Step:04
Game Logic:

```python
#Game Logic:
def next_turn(snake, food):
    # Update snake's position and handle collisions
    x, y = snake.coordinates[0]

    if direction == "up":
        y -= SPACE_SIZE
    elif direction == "down":
        y += SPACE_SIZE
    elif direction == "left":
        x -= SPACE_SIZE
    elif direction == "right":
        x += SPACE_SIZE

    snake.coordinates.insert(0, (x, y))

    square = canvas.create_rectangle(x, y, x + SPACE_SIZE, y + SPACE_SIZE, fill=SNAKE)
    snake.squares.insert(0, square)

    if x == food.coordinates[0] and y == food.coordinates[1]:
        global score
        score += 1
        label.config(text="Points:{}".format(score))
        canvas.delete("food")
        food = Food()
        snake.increase_size()  # Increase the size of the snake
    else:
        del snake.coordinates[-1]
        canvas.delete(snake.squares[-1])
        del snake.squares[-1]

    if check_collisions(snake):
        game_over()
    else:
        food.move()  # Move the food
        window.after(SPEED, next_turn, snake, food)
```

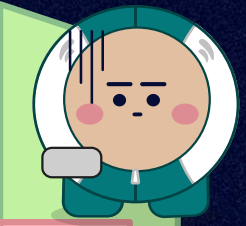


Step:05
Control Functions:

```python
#Control Functions:
def change_direction(new_direction):
    global direction, game_started
    if not game_started:
        game_started = True
        next_turn(snake, food)
    if new_direction == "left":
        if direction != "right":
            direction = new_direction
    elif new_direction == "right":
        if direction != "left":
            direction = new_direction
    elif new_direction == "up":
        if direction != "down":
            direction = new_direction
    elif new_direction == "down":
        if direction != "up":
            direction = new_direction
def check_collisions(snake):
    # Check for collisions with the snake's body or boundaries
    x, y = snake.coordinates[0]
    if x < 0 or x >= WIDTH:
        return True
    elif y < 0 or y >= HEIGHT:
        return True
    for body_part in snake.coordinates[1:]:
        if x == body_part[0] and y == body_part[1]:
            return True
    return False
def game_over():
    # Handle game over scenario
    canvas.delete(ALL)
    canvas.create_text(
        canvas.winfo_width() / 2,
        canvas.winfo_height() / 2,
        font=("consolas", 70),
        text="GAME OVER",
        fill="red",
        tag="gameover",
    )
```

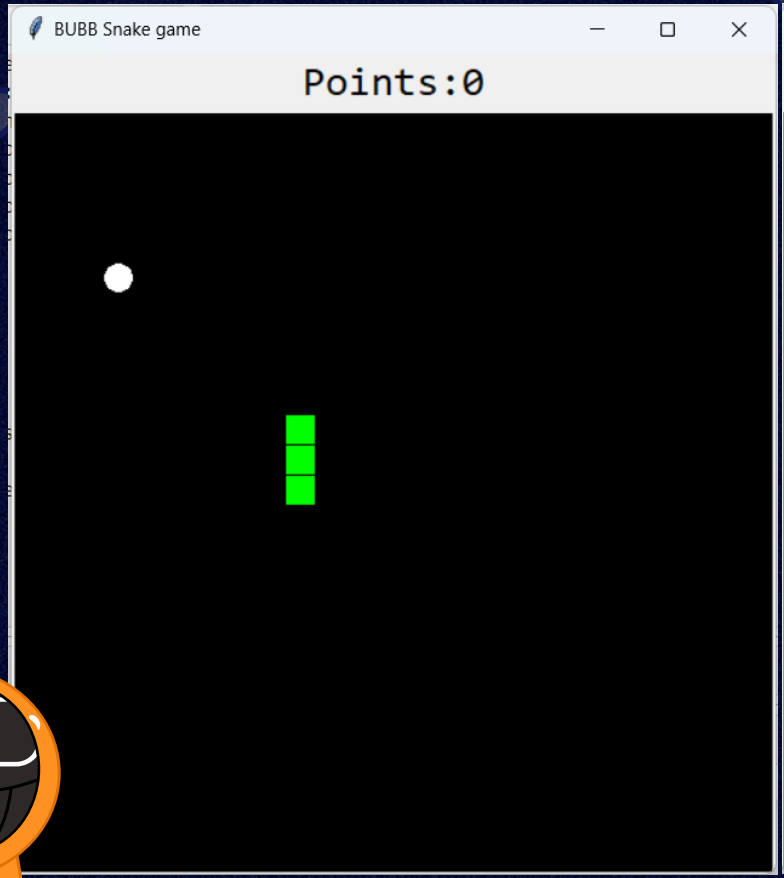


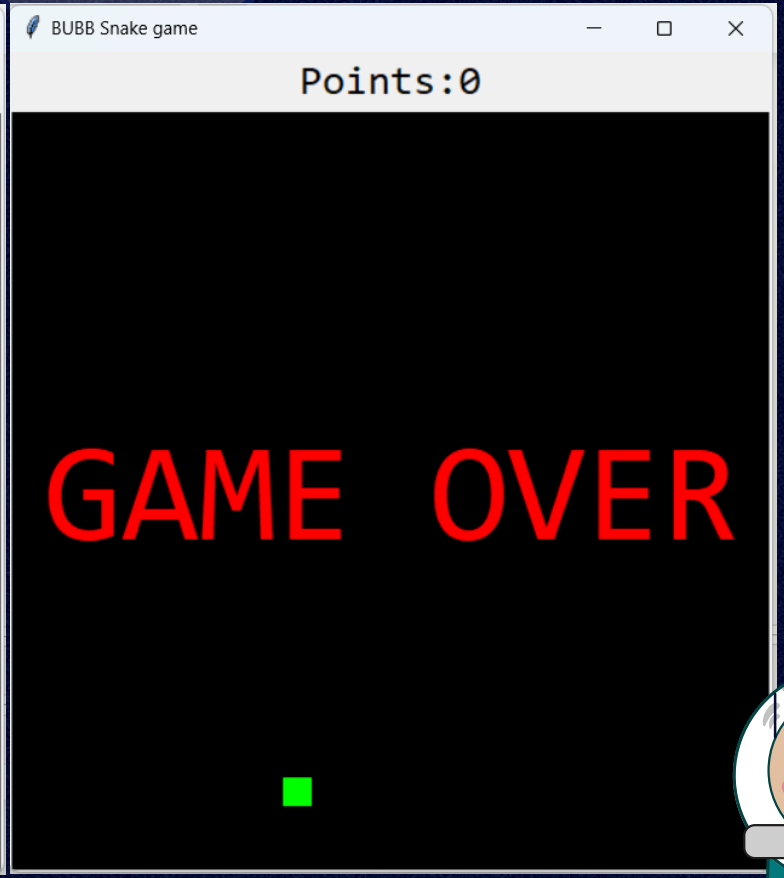
Step:06
Game Window and Key Bindings:

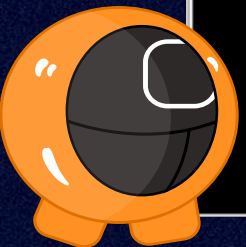


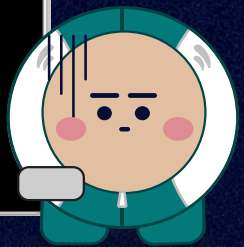
```
#Game Window and Key Bindings:
window = Tk()
window.title("GFG Snake game")
score = 0
direction = "down"
game_started = False
label = Label(window, text="Points:{}".format(score), font=("consolas", 20))
label.pack()
canvas = Canvas(window, bg=BACKGROUND, height=HEIGHT, width=WIDTH)
canvas.pack()
window.update()
window_width = window.winfo_width()
window_height = window.winfo_height()
screen_width = window.winfo_screenwidth()
screen_height = window.winfo_screenheight()
x = int((screen_width / 2) - (window_width / 2))
y = int((screen_height / 2) - (window_height / 2))
window.geometry(f"{window_width}x{window_height}+{x}+{y}")
window.bind("<Left>", lambda event: change_direction("left"))
window.bind("<Right>", lambda event: change_direction("right"))
window.bind("<Up>", lambda event: change_direction("up"))
window.bind("<Down>", lambda event: change_direction("down"))
window.bind("j", lambda event: food.change_direction("left"))
window.bind("l", lambda event: food.change_direction("right"))
window.bind("i", lambda event: food.change_direction("up"))
window.bind("k", lambda event: food.change_direction("down"))
snake = Snake()
food = Food()
def increase_snake_size():
    snake.increase_size()
    window.after(5000, increase_snake_size)
window.after(5000, increase_snake_size)
window.mainloop()
```

Results.









Thank You,
Everyone,