Simple Snake Game

Game description: A snake game is a simple game where a snake constantly moves around a box trying to eat an apple. Once it successfully eats the apple, the length of the snake increases. The game is over when the snake runs into itself or any of the four walls of the box.

Example: See https://playsnake.org/ for a simple example of the game.

Implement Snake Game according to the following rules:

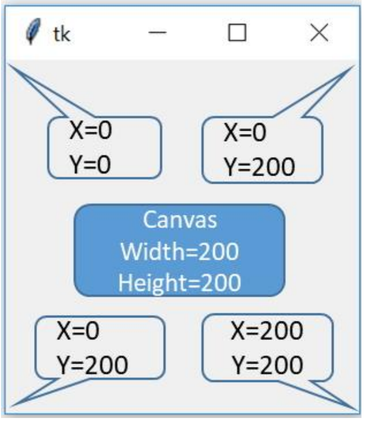
- Use **Tkinter** to create a **400x400 canvas widget** and a **score label**, initialized at 0.
- Each cell for the snake's body should be **20x20** pixels. The same applies for the food graphic.
- The game starts with a moving snake of **length two** at the **top left corner** of the canvas, initially heading to the **right direction**, and a food at a random position.
- Through the game, the snake's coordinates should be updated after each 300 milliseconds. (hint: use "after" method, see ColorGame example from the course slides).
- Using "up", "down", "left", "right" arrows on the keyboard, the player can change the direction of the moving snake if <u>not</u>:
  - The snake is heading toward "up" and user presses "down" arrow.
  - The snake is heading toward "down" and user presses "up" arrow.
  - The snake is heading toward "left" and user presses "right" arrow.
  - The snake is heading toward "right" and user presses "left" arrow.
- If the snake successfully hits a "food" on the board, the score increases by one (the label is updated), the length of the snake increases by one and the food is replaced by another one at a different random location in the canvas.
- Continue till the snake runs into the four surrounding walls of the canvas or itself. Display "Game over" in the middle of the cleared canvas when the player loses.

A suggestion for the object-oriented design of the game is provided at the end of the file.

Important! The game should be implemented using OOP concepts. Procedural implementations are not accepted. Do not use global variables.

<u>H.1. Canvas widget in tkinter:</u>

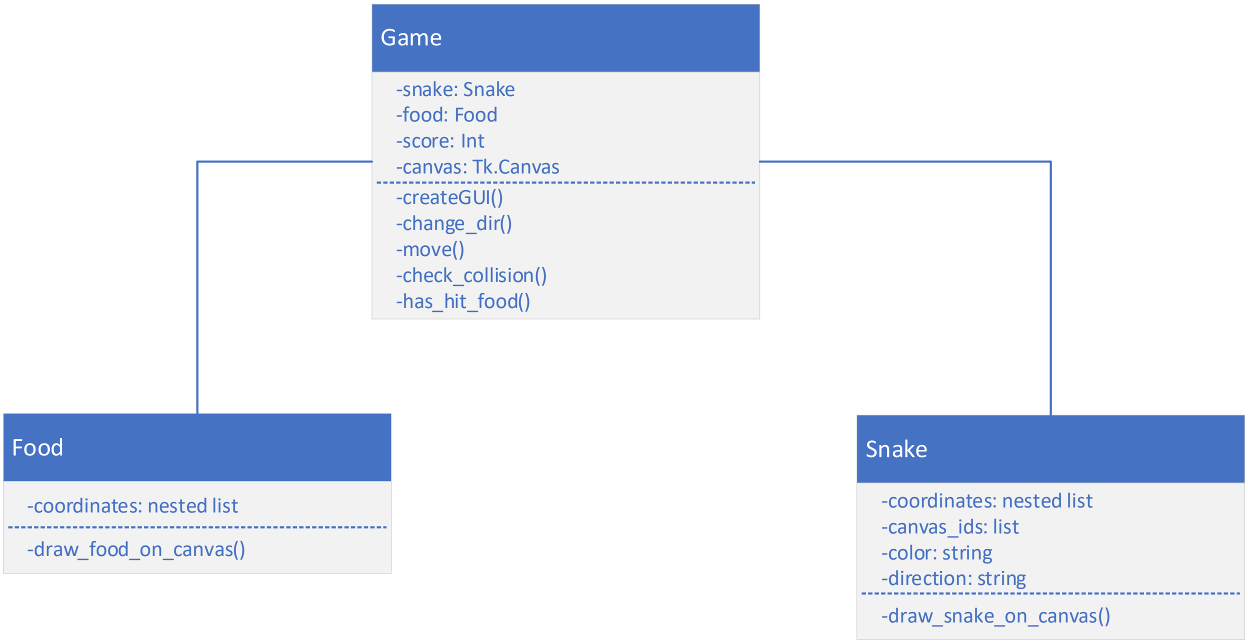
The Canvas is a *rectangular area* intended for drawing graphs such as lines, rectangles, text, ovals and polygons. Canvas has a coordinate system composed of (x,y) to place other objects/widgets at any given location. The coordinate system of canvas is upside down from what you might expect mathematically. The x coordinates increase from left to right, but the y coordinates increase from top to bottom. So the origin (0,0) is in the top left corner of the window. Here is an example of coordinates in a canvas.



In this homework, you will only use following methods of this widget:

- *Creating canvas:* canvas = tk.Canvas(root, width=,height=,bg=)
- *Drawing a rectangle on the canvas:* id=create_rectangle(x0, y0,x1,y1,fill=,tag=)
- *Drawing an oval on the canvas:* id=create_oval(x0,y0,x1,y1,fill=,tag=)
- *Writing a text on the canvas:* canvas.create_text(x, y, text=, fill=, font=)
- *Delete object/drawing from the canvas:*
  - canvas.delete("all") – this method clears everything.
  - canvas.delete(tag) – this method deletes the object corresponding to the given tag.
  - canvas.delete(ids) – this method deletes all the objects with the corresponding ids.

H.2. OOD of Snake Game:


Game
-snake: Snake
-food: Food
-score: Int
-canvas: Tk.Canvas
-createGUI()
-change_dir()
-move()
-check_collision()
-has_hit_food()
Food
-coordinates: nested list
-draw_food_on_canvas()
Snake
-coordinates: nested list
-canvas_ids: list
-color: string
-direction: string
-draw_snake_on_canvas()