Design Overview for Snake Game

Name: Ye Myat Kyaw
Student ID: J23039099

Summary of Program

The program will be the recreation of the classic snake game in which, the snake grows as it eats the fruits. The game ends when the snake hits itself or the borders.

Required Data Types

Describe each of the records and enumerations you will create using the following table (one per record).

Table 1: <<Snake>> details

Field Name	Type	Notes
Snake_x	Integer	X-coordinate of the snake
Snake_y	Integer	Y-coordinate of the snkae
Move_x	Integer	Horizontal increment
Move_y	Integer	Vertical increment
tail	integer	Length of snake tail
position	Array	Each coordinate of the snake's body

Table 2: <<Fruit>> details

Field Name	Type	Notes
Fruit_x	Integer	X-coordinate of the fruit
Fruit_y	Integer	Y-coordinate of the fruit

Table 3: <<ScreenType>> details

Value	Notes
MAIN_MENU	Main Menu Screen
GAME	Screen for the Snake game
GAMEOVER	Display when gameover

…

Overview of Program Structure

SnakeGame Class

1. **initialize**

Initializes the game window, sets up initial game state, loads assets, and initializes the snake, fruit, and background.

2. **update**

Updates the game logic, including the snake's movement, fruit interaction, and score management.

3. **draw**

Draws the current game screen based on the game state, including the main menu, game screen, and game over screen.

4. **draw_main_menu**

Draws the main menu, including the start button and background cover.

5. **draw_game_screen**

Draws the game screen, including the snake, fruit, and score display.

6. **draw_game_over**

Draws the game over screen, including the score and restart instructions.

7. **draw_score**

Draws the current score and highest score at the top of the game screen.

8. **button_down**

Handles keyboard input for game control, including snake movement and game state changes.

9. **reset_game**

Resets the game state for a new game, including the score, snake, and fruit positions.

10. **game_over**

Handles game over state, updating the highest score if necessary and switching to the game over screen.

11. **mouse_on_area?**

Checks if the mouse is within a specified area, used for detecting clicks on the start button.

Snake Class

1. **initialize**

Initializes the snake's position, movement, tail length, and loads head images.

2. **update**

Updates the snake's position, checks for collisions, and manages the snake's tail.

3. **draw**

Draws the snake's head facing the correct direction and the body segments.

4. **grow**

Increases the snake's tail length when it eats a fruit.

5. **moving_horizontal?**

Returns true if the snake is currently moving horizontally.

6. **up**

Changes the snake's direction to up if it is currently moving horizontally.

7. **down**

Changes the snake's direction to down if it is currently moving horizontally.

8. **left**

Changes the snake's direction to left if it is currently moving vertically.

9. **right**

Changes the snake's direction to right if it is currently moving vertically.

Fruit Class

1. **initialize**

Initializes the fruit's position.

2. **update**

Updates the fruit's state, if needed.

3. **draw**

Draws the fruit on the screen.

4. **generate_fruit**

Generates a new fruit position that does not overlap with the snake's body.

5. **overlap_fruit_snake?**

Checks if a new fruit position overlaps with the snake's body.

BackgroundPixel Class

1. **initialize**

Initializes the background pixels' positions.

2. **draw**

Draws the background grid.

Additional Functions

1. **play_eat_sound**

Plays a sound effect when the snake eats a fruit.

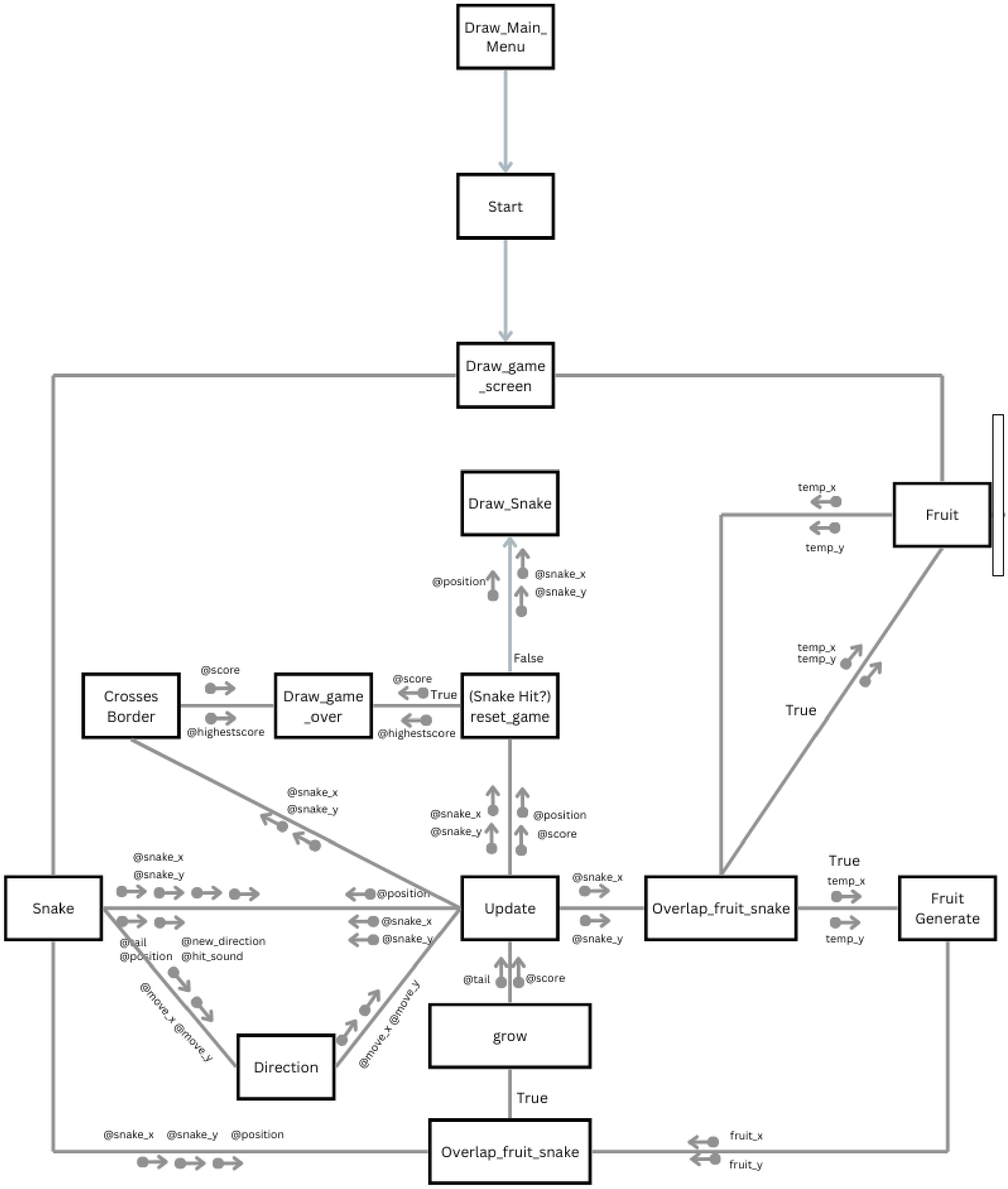
2. **play_background_music**

Plays background music during the game.

This list covers the main functions and procedures needed for the core functionality of the Snake game.

Custom Program Design

Structure Chart


Draw_Main_ Menu
Start
Draw_game _screen
Draw_Snake
Fruit
temp_x
temp_y
@position
@snake_x
@snake_y
False
temp_x
temp_y
True
Crosses Border
@score
@highestscore
Draw_game _over
@score
True
@highestscore
(Snake Hit?) reset_game
@snake_x
@snake_y
@snake_x
@snake_y
@position
@score
@snake_x
@snake_y
Snake
@position
@snake_x
@snake_y
Update
@snake_x
@snake_y
Overlap_fruit_snake
True
temp_x
temp_y
Fruit Generate
@tail
@position
@new_direction
@hit_sound
@tail
@score
@move_x @move_y
@move_x @move_y
Direction
grow
True
@snake_x
@snake_y
@position
Overlap_fruit_snake
fruit_x
fruit_y