**Snake Module Explanation**

**Class Interaction**

***Abutton***: This class creates a button, each button has a color and a string written on it. The Abutton class is used in the SnakeGame class in order to create buttons.

***Alarm***: Creates a thread which a program can used to create time based events. The Alarm class is used in SnakeGame in order to run certain methods every second. These methods create the snake and update the location of the snake every delay period.

***AlarmInterface***: Defines the methods used to interact with an Alarm. This interface is used in the Alarm class.

***Client***: Runs the game of Snake. This class creates an instance of the SnakeGame class, which creates the game.

***Collection***: Stores the linked list that the snake is made up of. Collection is used in the SnakeGame class in order to store the contents of each square on the screen. This is manipulated by the collection class to determine collisions and sort the linked list.

***CollectionInterface***: Interface to create a collection of a linked list for the game Snake. CollectionInterface is implemented in the Collection class.

***Node***: This class creates a node for the linked list. Node is used in the Collection class. It provides node operations used on the collection class's linked list.

***SnakeGame***: Creates the game environment for the game snake. SnakeGame is used in the Client class in order to run the game. The Abutton class is used to create interactive buttons. The Alarm class is used to create a thread so that the snake updated each delay period. The Collection class is used in SnakeGame to create the linked list of each square in the snake. The square SquareHead class is used in SnakeGame in order to create the beginning of the snake. The SquareItem class is used in SnakeGame to create the item that the snake eats. Finally, UneFenetre is used to allow window closing events.

***SnakeGameWithSidePanel*** : Creates the side panel of the game. This class extends SnakeGame, it runs the game of snake as well as everything in the side panel. There was so many constants in the side panel that I decided to make a separate class containing them.

***Square***: Creates a basic square. This class is the parent class of SquareFollower, SquareHead, and SquareItem. This class is used in the Collection class as a parameter for a few methods. The square is received to be manipulated by the linked list.

***SquareFollower***: Creates a square with the sole intention to follow the head square. This class is used in the SnakeGame class in order to create squares following the head.

***SquareHead***: Creates a square that can move. This class is used in SnakeGame in order to create the head of the snake.

***SquareItem***: Creates a square with a value. SquareItem is used in the SnakeGame class in order to create an item.

***UneFenetre***: Allows for the window to close. This is created in the SnakeGame class in order to allow the window to close.

**Specifications**

***Dynamic Structure***: A linked list is used in order to store the x and y location and color of each square. An additional link is added each time a square is added to the back of the snake. The links hold the x and y locations of the squares. The nodes are then be manipulated appropriately by the Collection class.

***Sorting Operation***: A bubblesort operation is used to sort the linked list according to the colors of each link in the snake. Colors are sorted based on their place in the hues that make a rainbow (roygbiv).

***Non-Trivial Class Inheritance***: Class inheritance is achieved by creating a Square class. The square class creates a basic square. That square is then used for the SquareHead, SquareFollower, and SquareItem class. In SquareHead, the square is enabled to have different movement directions. The SquareFollower class allows the square to follow a square from the SquareHead class. The SquareItem class takes the square and gives it a score value. This square is used in the game to be collected and to increment the score.

***Interaction***: Interaction in the game is available through the buttons located in the game. Also key inputs can be used in the game. See The user manual for more details.

***Graphical User Interface***: The main game appears in a rectangular box. To the side of the box the buttons and score are shown. The user can interact with buttons on the side panel. See the user manual for more details.