Virtual Piano and MIDI file player

* Project Functionality:
  + This code will create a virtual piano GUI that will play piano tones based on which key the user hovers over. An animation will occur for each key graphic that is played to display that it is currently in an action that plays a tone. The code will also have a function to load and play a MIDI file given by a user through a system browser. When a file is selected and the play button is pressed. The virtual piano will start playing the MIDI file through piano tones and press and animate the corresponding keys on the piano. Users should have the ability to start and stop MIDI file and resume playing at any given time.
  + Extra function if time is available:
    - Animate a graphic that displays incoming notes by a scrolling graphic of rectangles in the same x-axis as their corresponding note keys and their duration as the rectangles’ length. The piano will then play the key when a rectangle hits its corresponding key. (Think guitar hero or Synthesia)
* Classes
  + Main\_Interface extends JFrame
    - Uses:
      * MidiSequencer, Notes, KeyboardPanel, and Key classes
    - Function:
      * Creates the JFrame in main. Will house the functions to load (Create a new MidiSequencer), play, and pause the current midi file being played.
      * Sets up the interface with the KeyboardPanel class in a borderLayout
      * Will hold a thread to keep track of the current time to be able to start and stop a midi player
  + KeyboardPanel extends JPanel
    - Uses:
      * Key class
    - Function:
      * Creates a visual of Key objects(Rectangle objects) by loading all the white key pitches and black key pitches when creating Key objects with pitch integer values as a parameter during creation. The white and black keys will be stored in different Key Array so that the constructor of the KeyboardPanel can dynamically space and position white and black keys according to their pitch values. Then, it will add the keys from both arrays into a single array, keys, with indexes corresponding to the pitches of the notes (Highest pitch will be index 87, Lowest pitch will be index 0).
      * A mouseMotionListener will then be added to call a play function of the key the user hovers over on the GUI. This will check if a new key is hovered over and if so, plays the corresponding key’s pitch value. As will, this will implement a receiver function for when the Main\_Interface selects play and reads through the MIDI notes, it can transmit an event to make a key on the keyboard be played. This would also animate the current key being played.
  + Key extends Rectangle
    - Function:
      * The class is used to represent the keys on a piano as individual rectangles that will hold a pitch value. It will also contain function that will turn on and off the note’s sound (on(), off()). This is so the playMidi() function in Main\_Interface can easily turn on and off notes based on the Note class’s start and duration variables.
  + MidiSequencer
    - Uses:
      * Notes Class
    - Function:
      * This class will use file input to take a MIDI file from a designation and convert it into an ArrayList<ArrayList<Notes>> which will be called tracks. Since, MIDI files have the capability to store more than one track, a two dimensional array list is needed. Then, the class will be able to convert the two dimensional ArrayList into a HashMap<int, ArrayList<Notes> > which the key is the starttime value (in Notes class) and the object is an ArrayList of the Note object from track that have the same start values. This is so that the playMidi function can easily access if there are notes on the current time and how many notes are there to be played. This would solve the issue of having to search through multiple tracks to try and find a specific note that starts at this specific time.
  + Notes
    - Function:
      * Stores the values read in by a MIDI file including the channel, duration, pitch, startTime, and velocity of a note event. This is used to identify the attributes once a note is played.

