```
// SETUP ============================================
// Set up speaker on a PWM pin (digital 9, 10 or 11)
int speakerOut = 9;
// Do we want debugging on serial out? 1 for yes, 0 for no
int DEBUG =1;
char val; // Data received from the serial port
void setup() {
  // TONES  ==========================================
// Start by defining the relationship between
//       note, period, &  frequency.
#define  c     3830    // 261 Hz
#define  d     3400    // 294 Hz
#define  e     3038    // 329 Hz
#define  f     2864    // 349 Hz
#define  g     2550    // 392 Hz
#define  a     2272    // 440 Hz
#define  b     2028    // 493 Hz
#define  C     1912    // 523 Hz
// Define a special note, 'R', to represent a rest
#define  R     0
  pinMode(speakerOut, OUTPUT);
  if (DEBUG) {
    Serial.begin(9600); // Set serial out if we want debugging
  }
}

// MELODY and TIMING  =======================================
//  melody[] is an array of notes, accompanied by beats[],
//  which sets each note's relative length (higher #, longer note)
int melody[] = {  c,d,e,f,g,a,b,C };
int tune[] = { g, c, d, e, f, g, c, c, a, f, g, a, b, C, c, c};
int beats[]  = { 32, 16, 16, 16,  16,  32, 32, 32,32,16,16,16,16,32,32};
int TUNE_COUNT = sizeof(tune) / 2; // Melody length, for looping.

// Set overall tempo
long tempo = 10000;
// Set length of pause between notes
int pause = 1000;
// Loop variable to increase Rest length
int rest_count = 100; //<-BLETCHEROUS HACK; See NOTES

// Initialize core variables
int tone_ = 0;
int beat = 0;
long duration  = 0;

// PLAY TONE  ==============================================
// Pulse the speaker to play a tone for a particular duration
void playTone() {
  long elapsed_time = 0;
  if (tone_ > 0) { // if this isn't a Rest beat, while the tone has
    //  played less long than 'duration', pulse speaker HIGH and LOW
    while (elapsed_time < duration) {

      digitalWrite(speakerOut,HIGH);
      delayMicroseconds(tone_ / 2);

      // DOWN
      digitalWrite(speakerOut, LOW);
      delayMicroseconds(tone_ / 2);

      // Keep track of how long we pulsed
      elapsed_time += (tone_);
    }
  }
  else { // Rest beat; loop times delay
    for (int j = 0; j < rest_count; j++) { // See NOTE on rest_count
      delayMicroseconds(duration);
    }
  }
}

void loop() {
   if (Serial.available()) { // If data is available,
     tone_=0;
     int i= 0;
      val = Serial.read(); // read it and store it in val
      Serial.print(val);
      if (val == 'e' ) {
         i=0;
         tone_ = melody[i];
         beat = 32;
         duration = beat * tempo; // Set up timing
         playTone();
        // A pause between notes...
        delayMicroseconds(pause);
      } if (val == 'r' ) {
         i=1;
         tone_ = melody[i];
         beat = 32;
         duration = beat * tempo; // Set up timing
         playTone();
        // A pause between notes...
        delayMicroseconds(pause);
      } if (val == 't' ) {
         i=2;
         tone_ = melody[i];
         beat = 32;
         duration = beat * tempo; // Set up timing
         playTone();
        // A pause between notes...
        delayMicroseconds(pause);
      } if (val == 'y' ) {
         i=3;
         tone_ = melody[i];
         beat = 32;
         duration = beat * tempo; // Set up timing
         playTone();
        // A pause between notes...
        delayMicroseconds(pause);
      } if (val == 'u' ) {
         i=4;
         tone_ = melody[i];
         beat = 32;
         duration = beat * tempo; // Set up timing
         playTone();
        // A pause between notes...
        delayMicroseconds(pause);
      } if (val == 'i' ) {
           i=5;
           tone_ = melody[i];
            beat = 32;
           duration = beat * tempo; // Set up timing
           playTone();
          // A pause between notes...
          delayMicroseconds(pause);
      } if (val == 'o' ) {
           i=6;
           tone_ = melody[i];
           beat = 32;
           duration = beat * tempo; // Set up timing
           playTone();
          // A pause between notes...
          delayMicroseconds(pause);
      } if (val == 'p' ) {
           i=7;
           tone_ = melody[i];
           beat = 32;
           duration = beat * tempo; // Set up timing
           playTone();
          // A pause between notes...
           delayMicroseconds(pause);
       } else if (val == 'w') {                      // Play the pre programmed
tune
           for (i = 0; i<TUNE_COUNT;i++) { // Loop over the tune
                tone_ = tune[i];
                beat = beats[i];
                duration = beat * tempo; // Set up timing
                 playTone();
                 // A pause between notes...
                delayMicroseconds(pause);
           }
        }
      //tone_ = melody[i];



     if (DEBUG) { // If debugging, report loop, tone, beat, and duration
        Serial.print(val + " ");
        Serial.print(i);
        Serial.print(":");
        Serial.print(beat);
        Serial.print(" ");
        Serial.print(tone_);
        Serial.print(" ");
        Serial.println(duration);
      }
    }
}
```