```
/*
"rebound" 1 player version
author: John Hamm
created 4.2.15

build 07 - finalize scoring + display

build 06 - add RGB LED with result feedback
  + correct button being pressed for period of time (ignore after 1st press)
  + start sequence
  + add "great" zone/alter "good" zone
  + add serial feedback of zone + speed

build 05 - 1 player game setup (no brightness controls of LED runs)
*/

#include <ShiftRegister.h>    //uses shiftRegisterLib from Leonardo Banderali

ShiftRegister sr(1, 8, 10, 9, 12);
/*
-parameters:
  1 shift register
  pin 8 -> data
  pin 10 -> clock
  pin 9 -> latch
  pin 12 -> reset
*/

int buttonApin = 13;

int ballSpeed = 300;    //initial time between ball positions
int speedIncrease = 0;    //factor to increase speed

int score=0;          //scoring value

int buttonPress = 0;    //track if button pushed in time 0=no, 1=good, 2=great

int earlyZone = 50;    //% of ballSpeed interval where button press in rebound zone is too early
int goodZone = 25;    //% of ballSpeed interval where button press in rebound zone is good
```

```
int greatZone = 25;    //% of ballSpeed interval where button press in rebound zone is great (early + good + great=100%=interval between led 0 and home)
int lateZone = 50;    //% of ballSpeed interval where button press in rebound zone is late, but accepted

int redHomePin = 3;      //pin for homeRGB LED connected to red
int greenHomePin = 5;    //pin for homeRGB LED connected to red
int blueHomePin = 6;      //pin for homeRGB LED connected to red

int scoreTens = 0;    //final scoring tabulation
int scoreOnes = 0;
int counter = 0;    //for score display calculation


//------------------------------setup---------------------------------

void setup() {

  pinMode(buttonApin, INPUT_PULLUP);

  Serial.begin(9600);
   sr.setShiftOrder(MSBFIRST);  //set order in which bits are shifted (MSBFIRST/LSBFIRST)
  sr.clear();                  //set all bits to low

  while (! Serial); // Wait untilSerial is ready
  Serial.println("Ready to start...");        //serial output used throughout to verify gameplay

  allOff();
  delay(1000);
  startSeq();    //start sequence - home LED blinks purple 3x, then green once
}

//------------------------------game loop-------------------------------------

void loop() {

  Serial.print("ball speed is ");
  Serial.println(ballSpeed);
  runUpDown();
  reboundCheck();
  if (buttonPress == 0) {
    endGameLate();
```

```
  }
  if (buttonPress == 1) {
    ballSpeedIncrease();
  }
  if (buttonPress == 2) {
    ballSpeedDecrease();
  }
}


//------------------functions--------------------------

//cycle of LEDs from start to end (0-7) and back (6-1) with check for early button push
void runUpDown() {
   buttonPress=0;
   for (int i = 0; i < 8; i++) {
     for (int k = 0; k < 100; k++) {
       sr.setBit(i, HIGH);
       checkButtonBad();
       delay(ballSpeed / 100);
     }
    sr.setBit(i, LOW);
  }
  allOff();    // to turn off home RGB LED feedback from previous rebound
  for (int i = 6; i > 0; i--) {
    for (int k = 0; k < 100; k++) {
      sr.setBit(i, HIGH);
      checkButtonBad();
      delay(ballSpeed / 100);
    }
    sr.setBit(i, LOW);
  }
}


//check bad button press during runUpDown + early rebound
void checkButtonBad() {
    if (digitalRead(buttonApin) == LOW) {
      Serial.println("too soon! ");
      endGameEarly();
```

```
    }
  }


  //check for button press in the rebound "zone"
  void reboundCheck() {
   for (int k = 0; k < earlyZone; k++) {    //check 100x/interval for early button press 1st % of interval time
       sr.setBit(0, HIGH);
       checkButtonBad();
       delay(ballSpeed / 100);
     }
   for (int k = 0; k < goodZone; k++) {    //check for early button press in good % of interval time
       if (buttonPress == 0) {
         checkButtonGood();
        }
       delay(ballSpeed / 100);
     }
   for (int k = 0; k < greatZone; k++) {    //check for early button press in great % of interval time
       if (buttonPress == 0) {
         checkButtonGreat();
        }
       delay(ballSpeed / 100);
     }
   sr.setBit(0, LOW);    //turn off 1st ball
   for (int k = 0; k < lateZone; k++) {    //check for late button press % after
       if (buttonPress == 0) {
         checkButtonLate();
        }
       delay(ballSpeed / 100);
     }
  }


  //check button press during "good" zone
  void checkButtonGood() {
     if (digitalRead(buttonApin) == LOW) {
       buttonPress=1;
       score = score + 1;
       analogWrite(greenHomePin, 120);
       analogWrite(blueHomePin, 0);
```

```
      Serial.println("good! ");
      Serial.print("score: ");
      Serial.println(score);
    }
  }

  //check button press during "great" zone
  void checkButtonGreat() {
    if (digitalRead(buttonApin) == LOW) {
      buttonPress=2;
      score = score + 3;
      analogWrite(greenHomePin, 0);
      Serial.println("great! ");
      Serial.print("score: ");
      Serial.println(score);
    }
  }



  //check button press during "late" zone
  void checkButtonLate() {
    if (digitalRead(buttonApin) == LOW) {
      buttonPress=1;
      analogWrite(greenHomePin, 120);
      analogWrite(redHomePin, 120);
      Serial.println("late, but still good! ");
    }
  }



  void ballSpeedIncrease() {
   speedIncrease = ballSpeed / 10;       //increase ballSpeed by 10% if in good zone, or late zone
   ballSpeed = ballSpeed - speedIncrease;
  }

  void ballSpeedDecrease() {
   speedIncrease = ballSpeed / 10;
   ballSpeed = ballSpeed + speedIncrease;     //decrease ballSpeed by 10% if in good zone, or late zone
  }
```

```
//button pressed too early
void endGameEarly() {
  Serial.print("Game over. Final score: ");
  Serial.println(score);
  scoreDisplay();
}

//button not pressed in time
void endGameLate() {
  Serial.println("too late sucker!");
  Serial.print("Game over. Final score: ");
  Serial.println(score);
  scoreDisplay();
}

//turns all LEDs off
void allOff() {
  for (int i=0; i<8; i++) {
    sr.setBit(i, LOW);
    analogWrite(redHomePin, 255);
    analogWrite(greenHomePin, 255);
    analogWrite(blueHomePin, 255);
  }
}

//signify end of game, then display score
void scoreDisplay() {
  for (int k=0; k<10; k++){      //flash all LEDs + home Red 10x
    for (int i=0; i<8; i++) {
      sr.setBit(i, HIGH);
    }
    analogWrite(redHomePin, 0);
    delay(50);
    allOff();
    delay(50);
  }
```

```
    delay (1000);

    for (int k=1; k<10; k++) {    //calculate how many "tens" and "ones" in score
       counter=k*10;
       if (score > counter -1) {
       scoreTens = counter/10;
       scoreOnes = score - counter;
       }
       if (score < 10) {
         scoreOnes = score;
       }
    }
    Serial.print("scoreTens: ");
    Serial.println(scoreTens);
    Serial.print("scoreOnes: ");
    Serial.println(scoreOnes);

    for (int k = 0; k <3; k++) {          //display score 3x
      for (int i=0; i<scoreTens; i++) {    //display "tens" as green home flashes
        analogWrite(greenHomePin, 0);
        delay(500);
        allOff();
        delay(500);
      }
      if (scoreOnes > 0) {
        for (int i=7; i>7-scoreOnes; i--) {    //display "ones" as red leds from 8 to 0
          sr.setBit(i, HIGH);
          if (i < 0) {
            analogWrite(redHomePin, 0);
          }
          delay (500);
        }
      }

      delay (1000);
      allOff();
      delay(500);
    }

    score = 0;        //reset values + start loop again
```

```
  ballSpeed = 300;
  scoreTens = 0;
  scoreOnes = 0;
  buttonPress = 0;
  Serial.println("Try again...");
  startSeq();
}


//start sequence
void startSeq() {
  for (int i = 0; i < 3; i++) {
    allOff();
    analogWrite(redHomePin, 0);
    analogWrite(blueHomePin, 0);
    delay (50);
    allOff();
    delay (ballSpeed - 50);
  }

  analogWrite(greenHomePin, 0);
  delay (50);
  allOff();
  delay (ballSpeed - 50);
}
```