```
//Get off your butt timer using LEDs and ATtiny 45
//Created by: Jonathan Bush
//Last Updated: 4/19/2017

//Schematic description:
//Reset button tied to pin 1 (reset) and pulled high. Reset when pin 1 sees low.
//Acknowledge/activity button tied to pin 2 (digital 3) and pulled high.
//Pin 4 (gnd) tied to GND
//Stand up LED tied to pin 6 (digital 1)
//Walk LED tied to pin 7 (digital 2)
//Pin 8 (Vcc) tied to 5v

#define WALK_LED 2 //Pin controlling LED behind the walking guy
#define STAND_LED 1 //Pin controlling LED behind the up arrow
#define BUTTON 3 //Pin reading acknowledge/activity button
#define BTN_DELAY 250 //To be used in our program to "debounce" the cheap buttons

int BTN_PRESSED = 0; //Define that the button is not initially pressed
unsigned long mark; //to track when button last pushed
unsigned long ONE_HOUR = 3600000; //one hour equals 1000 * 60 * 60 milliseconds

void setup(){
  pinMode(BUTTON, INPUT); //Define pin as an input
  pinMode(STAND_LED, OUTPUT); //Define pin as an output
  pinMode(WALK_LED, OUTPUT); //Define pin as an output

  //This for loop flashes the Stand/Walk LED back and forth to provide feedback that
the program has started over
  //Handy confirmation that the pressing the reset button was successful or that you
have power when initially plugged in
  for(int i=0;i<3;i++){
    digitalWrite(STAND_LED, HIGH);
    digitalWrite(WALK_LED, LOW);
     delay(500);
    digitalWrite(STAND_LED, LOW);
    digitalWrite(WALK_LED, HIGH);
     delay(500);
  }
  digitalWrite(STAND_LED, LOW); //turn both LEDs off before main part of code runs
  digitalWrite(WALK_LED, LOW);
}

void loop(){ //where all the super simplistic magic happens
  chkBtn(digitalRead(BUTTON));  //check to see if button is pressed every loop by
running this code
```

```
  //If it has been more than one hour since you walked, get up and walk around
  if((millis()-mark)> ONE_HOUR){ //calculation to see if it has ben greater than one
hour
    digitalWrite(WALK_LED,HIGH);
  }else{
    digitalWrite(WALK_LED,LOW);
  }

  //Stand after one hour, stand for one hour then sit for two, repeat all day.
  if(millis()>ONE_HOUR && millis()<(ONE_HOUR * 2)){
    digitalWrite(STAND_LED,HIGH);
  }else if(millis()>(ONE_HOUR * 4) && millis() < (ONE_HOUR *5)){
    digitalWrite(STAND_LED,HIGH);
  }else if(millis()>(ONE_HOUR * 7) && millis() < (ONE_HOUR *8)){
    digitalWrite(STAND_LED,HIGH);
  }else if(millis()>(ONE_HOUR * 10) && millis() < (ONE_HOUR *11)){
    digitalWrite(STAND_LED,HIGH);
  }else if(millis()>(ONE_HOUR * 13) && millis() < (ONE_HOUR *14)){
    digitalWrite(STAND_LED,HIGH);
  }else if(millis()>(ONE_HOUR * 16) && millis() < (ONE_HOUR *17)){
    digitalWrite(STAND_LED,HIGH);
  }else{
    digitalWrite(STAND_LED,LOW);
  }
}

boolean chkBtn(int buttonState) { //
  if (buttonState == LOW && (millis() - mark) > BTN_DELAY) { //check to see if
button has been pressed and that it is longer than delay to get rid of bouncing.
    mark = millis(); //set mark equal to current value of millis
    BTN_PRESSED = 1; //not needed, remnant from prior code I pulled this from.
    return true; //not needed, remnant from prior code I pulled this from.
  }
  else { return false; } //not needed, remnant from prior code I pulled this from.
}
```