```c
#include <wiringPi.h>  // For GPIO usage
#include <stdio.h>     // For standard I/O functions
#include <unistd.h>    // For sleep() and usleep()
#include <time.h>      // For time()

// GPIO pin connected to the buzzer
#define BUZZER_PIN 23

/**
 * @brief Plays a single tone on the buzzer.
 *
 * @param frequency Frequency of the tone in Hz
 * @param duration_ms Duration of the tone in milliseconds
 */
void playTone(int frequency, int duration_ms) {
    // Calculate the delay for half cycle in microseconds
    // 1 second = 1,000,000 microseconds, so half of 1 cycle
    // is 500,000 seconds
    int halfPeriod = 500000 / frequency;
    // Total cycles to generate the tone for
    int cycles = frequency * duration_ms / 1000;

    for (int i = 0; i < cycles; i++) {
        digitalWrite(BUZZER_PIN, HIGH); // Turn buzzer on
        usleep(halfPeriod);             // Wait half a period
        digitalWrite(BUZZER_PIN, LOW);  // Turn the buzzer off
        usleep(halfPeriod);             // Wait half a period
    }
}

/**
 * @brief Plays a cheerful jingle with 4 notes.
 *
 * Each note is followed by a short pause to improve clarity.
 * Without this pause, the notes bleed into one another.
 */
void playHappyChime() {
    playTone(262, 150);
    usleep(50000);
    playTone(330, 150);
    usleep(50000);
    playTone(392, 150);
    usleep(50000);
    playTone(523, 300);
}

/**
 * @brief Plays a sad jingle with 4 notes.
 *
 * Each note is followed by a short pause to improve clarity.
 * Without this pause, the notes bleed into one another.
 */
void playSadChime() {
    playTone(440, 300);
    usleep(50000);
    playTone(349, 300);
    usleep(50000);
    playTone(294, 300);
    usleep(50000);
    playTone(262, 400);
}

int main() {
    // Initialize GPIO pins
    wiringPiSetupGpio();
    // Set up the buzzer pin as the output
    pinMode(BUZZER_PIN, OUTPUT);

    // 0 = soil is dry, 1 = soil is moist
    int isMoist = 0;
    // The last time the happy chime was played
    time_t lastHappy = 0;
    // The last time the sad chime was played
    time_t lastSad = 0;

    // Infinite loop, since this program should theoretically always be running
    while (1) {
      // Get the current system time
        time_t now = time(NULL);

      // If the soil is moist and at least 10 seconds have passed, play the happy
chime
        if (isMoist && now - lastHappy >= 10) {
            playHappyChime();
            // Set the last time the happy chime was played to the current time
          lastHappy = now;
        }

      // If not moist and at least 10 seconds have passed, play sad chime
        if (!isMoist && now - lastSad >= 10) {
            playSadChime();
            // Set the last time the sad chime was played to the current time
          lastSad = now;
        }

      // Wait for 1 second before checking again
        sleep(1);
    }

    return 0;
}
```