4. Use the touch sensor to start a countdown on the 7-segment display. If the ultrasonic sensor detects an obstacle (within a specified range) during the countdown, reset the timer. Display "E" on the display if the countdown completes without interruption.

Program :

```
                         // Pin Definitions

const int trigPin = 9;

const int echoPin = 10;

const int touchPin = 11;

                         // 7-Segment Pins

const int segmentA = 2;

const int segmentB = 3;

const int segmentC = 4;

const int segmentD = 5;

const int segmentE = 6;

const int segmentF = 7;

const int segmentG = 8;

                 // 7-segment display number representation

const int numbers[10][7] = {

  {1, 1, 1, 1, 1, 1, 0},  // 0

  {0, 1, 0, 0, 0, 0, 0},  // 1

  {1, 1, 0, 1, 1, 0, 1},  // 2

  {1, 1, 0, 1, 0, 0, 1},  // 3

  {0, 1, 1, 0, 0, 0, 1},  // 4

  {1, 0, 1, 1, 0, 1, 1},  // 5
  {1, 0, 1, 1, 1, 1, 1},  // 6
  {0, 1, 0, 0, 0, 0, 0},  // 7
  {1, 1, 1, 1, 1, 1, 1},  // 8
  {1, 1, 1, 1, 0, 0, 1}   // 9
};
                                    // Distance threshold in cm
const long distanceThreshold = 20;
const int countdownTime = 10;     // Countdown time in seconds
void setup() {
                                            // Setup pins
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(touchPin, INPUT);
                                      // Setup 7-segment pins
  for (int i = 2; i <= 8; i++) {
    pinMode(i, OUTPUT);
  }
  Serial.begin(9600);
}
void loop() {
  if (digitalRead(touchPin) == HIGH) {
    startCountdown(countdownTime);
  }
}
```

```
void startCountdown(int time) {
  for (int i = time; i >= 0; i--) {
                                  // Check for obstacles during countdown
    if (isObstacleDetected()) {
      i = time;              // Reset timer if an obstacle is detected
    }
    displayNumber(i); // Display current countdown value
    delay(1000);         // Wait for one second
  }
  displayMessage("E");       // Display "E" after countdown
}
bool isObstacleDetected() {
  long distance = measureDistance();
  return (distance < distanceThreshold);
}
long measureDistance() {
                                          // Trigger the ultrasonic sensor
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
// Read the echo
  long duration = pulseIn(echoPin, HIGH);
                                              // Calculate distance in cm
  return duration * 0.034 / 2;
}
void displayNumber(int num) {
  if (num < 0 || num > 9) return;        // Display only single-digit numbers
                                      // Set segments based on the number
  for (int i = 0; i < 7; i++) {
    digitalWrite(i + 2, numbers[num][i]);
  }
}
void displayMessage(const char* message)
                // Assuming 'E' corresponds to a specific segment configuration
{
  int eSegments[7] = {1, 1, 1, 1, 1, 0, 1};   // "E" segment configuration
                                         // Display 'E' on the 7-segment
  for (int i = 0; i < 7; i++) {
    digitalWrite(i + 2, eSegments[i]);
  }
  delay(5000); // Display for 5 seconds

  clearDisplay(); // Clear the display
}
```

```
void clearDisplay() {
  // Turn off all segments
  for (int i = 0; i < 7; i++) {
    digitalWrite(i + 2, LOW);
  }
}
```