```c
#include <stdio.h>
#include <stdbool.h>
#include <math.h>

// Prototypes
double mag (double x, double y, double z);
double getDistanceFallen (double finalVelocity, double seconds);
bool closeTo (double tolerance, double target, double val);

int main () {

    bool fallen, falling;
    int dotI, exclamationI, thisTms;
    double ax, ay, az, acceleration, distanceFallen, fallTime, thisX, lastX, integralX, thisV,
    lastV, integralV, lastT, thisT, initialT;

    dotI = 0;
    exclamationI = 0;
    fallen = false;

    printf("OK, I'm now receiving data.\n");
    printf("I'm waiting");
    while (!fallen) {

        scanf("%d, %lf, %lf, %lf, %d, %d, %d, %d, %d, %d", &thisTms, &ax, &ay, &az);
        acceleration = mag(ax, ay, az);
        thisT = thisTms / 1000.0; // Converts milliseconds to seconds
        initialT = thisT;

        // Prints the right number of dots after "I'm waiting"
        if (dotI > 20) {
            printf(".");
            fflush(stdout);
            dotI = 0;
        }

        // The Esplora is falling if the magnitude of it's acceleration is no longer close to 1
        if (!closeTo(0.25, 1, acceleration)) {
            printf("\n\tHelp me! I'm falling");
            lastT = thisT;
            lastX = 0.0;
            lastV = 0.0;
            falling = true;

            while (falling) {

                // Print exclamations marks while the Esplora is falling
                if (exclamationI > 12) {
                    printf("!");
                    fflush(stdout);
                    exclamationI = 0;
                }
                // Scan in new variables and make any necessary conversions
                scanf("%d, %lf, %lf, %lf, %d, %d, %d, %d, %d, %d", &thisTms, &ax, &ay, &az);
                acceleration = mag(ax, ay, az);
                thisT = thisTms / 1000.0; // Converts milliseconds to seconds

                // Calculate the Riemann Sum of velocity over time to find distance function
                thisV = lastV + 9.8 * (1.0 - acceleration) * (thisT - lastT);

                // Calculate the Riemann Sum of the position over time to find the distance
                travelled
                thisX = lastX + thisV * (thisT - lastT);

                // Set the 'last' values for all the variables
                lastX = thisX;
                lastV = thisV;
                lastT = thisT;

                // Check to see if the Esplora is still falling
                if (closeTo(0.25, 1.0, acceleration)) {
                    falling = false;
                    fallen = true;
                }

                exclamationI++;

            }
        }
        dotI++;
    }

    fallTime = (lastT - initialT);
    distanceFallen = getDistanceFallen(acceleration, fallTime);
    printf("\n\t\tOuch! I fell %.3lf meters in %.10lf seconds, compensating for air
    resistance\n", thisX, fallTime);

    printf("\n\n\tCalculated distance using calculus:\t%lf meters in %lf seconds\n", thisX,
    fallTime);
    printf("\tCalculated distance using algebra:\t%lf meters in %lf seconds\n", distanceFallen,
    fallTime);
    printf("\tAlgebraic error: %lf meters (%lf%%)\n", distanceFallen - thisX, (distanceFallen -
    thisX) / thisX);

    return 0 ;

}

double mag (double x, double y, double z) {
    double magnitude = sqrt(pow(x, 2) + pow(y, 2) + pow(z, 2));
    return magnitude;
}

bool closeTo(double tolerance, double target, double val) {
    double lower, higher;
    lower = (target - tolerance);
    higher = (target + tolerance);
    if (val > lower && val < higher) {
        return true;
    } else {
        return false;
    }
}

double getDistanceFallen (double acceleration, double seconds) {
    return (0.5 * (9.8 - acceleration) * pow(seconds, 2)); // Using the equation d = .5*a*t^2
}
```