```
/********************************************************
 * monte_carlo: utilise a basic monte carlo method      *
 *              in order to find the volume of the      *
 *              cut of three cylinders with unit radius *
 *              along x, y and z                        *
 *                                                      *
 *              code example from lecture 4             *
 *              dated 08.05.19                          *
 ********************************************************/

#include <stdio.h>
#include <stdlib.h> // for random numbers
#include <time.h> // for seeding
#include <math.h> // don't forget to link with -lm

/*
In the following we implement a MC simulation. We want to generate
random vectors that are equally distributed in [-1,1]^3. For every
random vector generated, we check if it falls withing the body under
consideration. We save the number of total random vectors generated
as "i" and the number of times the body has been hit as "hits".

The formula for the approximated volume is then

volume = volume_random * (double) hits / i;

with volume_random the volume of the space in which the random vectors
live (here volume_random = 8).
*/

// total number of random vectors generated
#define MC_ITERATION_MAX 1000000

int main(){

    size_t i;
    size_t hits = 0;

    // seed random number generator
    srand(time(NULL));

    for(i = 0; i < MC_ITERATION_MAX; i++){
        // 3D random numbers
        double x = 2 * (double) rand() / RAND_MAX - 1; // -1 ... 1
        double y = 2 * (double) rand() / RAND_MAX - 1;
        double z = 2 * (double) rand() / RAND_MAX - 1;

        // if random vector inside body, increment hits
        if( x*x + y*y <= 1 && x*x + z*z <= 1 && y*y + z*z <= 1){
            hits ++;
        }
    }

    printf("Hits: %zu\n", hits);

    double volume_random = 8;
    double volume = volume_random * (double) hits / i;
    double volume_exact = 8 * (2 -sqrt(2));

    printf("Volume: %f\n", volume);
    // utilise fabs for floating point absolute value
    printf("Error: %f\n", fabs(volume - volume_exact) / volume_exact );
    return 0;
}
```

```

// Makefile (with linked math library)
/*
monte_carlo: monte_carlo.c
        gcc monte_carlo.c -o monte_carlo -lm
*/
```