```c
#include <math.h>
#include <time.h>
#include <stdio.h>
#include <stdlib.h>

double f(double x) // define function to be integrated
{
    return 3*x*x;
}

main()
{
    double x, a, b, fav, f2av, err, exact, nsigma;
    int i, NPOINT;

    printf ("NPOINT = ");
    scanf ("%d", &NPOINT);

    exact = 1;
    b = 1.0; // set upper limit
    a = 0; // set lower limit
    srand(time(NULL)); // initialize RNG

    printf ("<f(x)>   <f^2(x)> \n ");
    fav = 0; f2av = 0;
    for (i = 0; i < NPOINT; i++)
    {
        x = a + (b - a) * rand() / RAND_MAX; // random value for x
        fav += f(x); // compute f(x)
        f2av += f(x)*f(x); // compute f(x)^2
        // printf ("%10.5f    %10.5f %10.5f\n", x, f(x), f(x)*f(x));
    }
    fav /= NPOINT; // sample average of f
    f2av /= NPOINT; // sample average of f^2

    printf (" averages \%10.5f \%10.5f \%10.5f\n", fav, f2av);
    printf (" exact \%10.5f \%10.5f \%10.5f\n", b, a, 0);
    printf ("%10.5f \%10.5f \%10.5f \n", x, f(x), f(x)*f(x));

    printf (" err = sqrt((f2av - fav*fav) / (NPOINT - 1)); // error bar
    printf (" in answer = \%10.5f\n", (b - a) * fav);
    printf (" in error bar (sigma) = \%10.5f\n", (b - a) * err);
    printf (" n sigma = (fav - exact) / err; // no of st.devs. from exact
    printf (" in Note: 'error bar' = sqrt((<f^2(x)> - <f(x)>^2)/(NPOINT - 1))*\n");
}
```

This code defines a function to be integrated and uses Monte Carlo simulation to approximate the integral. It sets the limits of integration, initializes the random number generator, and samples points within the interval. The average of the function values and their squares are computed to estimate the integral. The error is calculated using the standard deviation of the samples.