One method to determine the value of \( \pi \) is to use the following Monte Carlo simulation and probability argument: Assume that \( N_{\text{Total}} \) points \((x_i, y_i)\) are randomly distributed in a unit square with \( 0 \leq x_i < 1.0 \) and \( 0 \leq y_i < 1.0 \). (See the picture below.) The probability of any random point being found inside the quarter unit circle, \( P_{\text{Inside}} \), is then proportional to ratio of the two areas, i.e.,

\[
P_{\text{Inside}} = \frac{A_{\text{Inside}}}{A_{\text{Square}}} = \frac{\pi/4}{1} = \frac{\pi}{4}
\]

(Eq. 1)

Therefore, for \( N_{\text{Total}} \) random points generated, we would expect to find the following number of points inside the quarter circle:

\[
N_{\text{Inside}} = N_{\text{Total}} \cdot P_{\text{Inside}} = N_{\text{Total}} \cdot \frac{\pi}{4}.
\]

(Eq. 2)

Your task is to write a Monte Carlo simulation that will generate a total of \( 10^6 \) such random points and then to count how many of these points fall inside the quarter circle by checking if such a point’s distance to the origin is less than the radius of the unit circle, i.e., \( 1 \). From knowing \( N_{\text{Total}} \) and \( N_{\text{Inside}} \) you should then be able to calculate the (guessed) value of \( \pi \) using equation 2 above.

1) (12.5 Points)

Write a complete console* ANSI-C program that calculates the guessed value of \( \pi \) as described above.

Hint: to generate a random number use the \textit{rand()} function; to ensure that it is in the range from 0 to 1 divide \textit{rand()} by \textit{RAND_MAX}.

Final Note: in your program do not include any input or output statements such as \textit{scanf, printf} or any include files. You will be graded on program logic and syntax mistakes. You may use local or global variables.

Console* means you must not include a LabWindows Graphical User Interface (GUI) or any (callback) function for it.
Solution:

```c
#include <ansi_c.h>
main()
{
    unsigned long i, inside=0, total = 10E6;
    double x, y,piguessed;
    char a;     //optional  - not required on test

    for( i = 0; i < total; i++)
    {
        x = (double) rand()/RAND_MAX;
        y = (double) rand()/RAND_MAX;
        if(x*x + y*y <= 1.0)
            inside++;
    }
    piguessed = 4.0*inside/total;
    printf("%lf",piguessed);    //optional  - not required on test
    scanf("%c", &a);            //optional  - not required on test
}
```