EECS 10: Assignment 5

October 20, 2006

Due Monday 10/30/2006 12:00pm

1 Compute the Greatest Common Divisor (gcd) [10 points]

Write a program in C to compute the greatest common divisor of two positive integers. In this progam, you are going to use the Euclid Algorithm to find the gcd of two positive integer number a, b:

```
Let gcd(a, b) be the greatest common divisor of a, b (where a>0,
b>0, and a >= b).

a) If a = k * b, where k is a positive integer, then gcd(a, b) = b.

b) If a = k * b + r, where k is a positive integer, and r is the
   remainder of a / b, then gcd(a, b) = gcd(b, r)
```

Examples:

```
1. To find gcd(10, 5)

   1) we observe that 10 = 2 * 5, which satisfies a). Therefore, gcd(10, 5) = 5.

2. To find gcd(12, 8)

   1) we observe that 12 = 1 * 8 + 4. According to b), gcd(12, 8) = gcd(8, 4).
      Now we want to find gcd(8, 4).

   2) we observe that 8 = 2 * 4, which satisfies a). Therefore, gcd(8, 4) = 4.
      Since gcd(12, 8) = gcd(8, 2), so gcd(12, 8) = 4.
```

When executed, your program should look like this:

```
Please Enter Two Positive Integers: 5 10
The GCD of 5 and 10 is: 5
```

You may assume that the inputs to the program are two positive integers. But the first input integer may be larger, smaller, or equal to the second input integer.

Compile your program and run it using the following three pairs of numbers: (25 12) (7 28) (12 28)

You should submit gcd.c, gcd.txt, and gcd.script for this problem.

2 Monte Carlo Calculation of Pi [20 points]

Monte Carlo (MC) methods are stochastic techniques–meaning they are based on the use of random numbers and probability statistics to investigate problems. In homework5, you are required to write a program to implement a simple geometric MC experiment which calculates the value of Pi based on a "hit and miss" integration.

The figure below shows a unit circle circumscribed by a square. The radius of the circle r equals to 1/2 of the side of the square. Furthermore, the center of circle and the center of squared are overlapped.

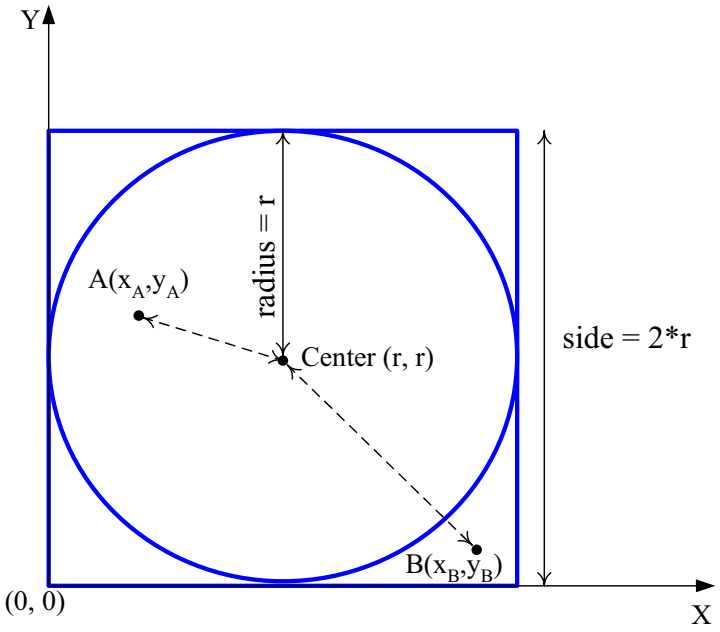

Figure 1: A Circle Circumscribed by a Square

Image we can throw points randomly at the above figure. If we can throw infinite random points, it should be apparent that of the total number of points that hit the circle, the number of points that hit within the square is proportional to the area of that part. In other words:

$\frac{number\ of\ points\ hitting\ circle\ area}{number\ of\ points\ hitting\ square\ area} = \frac{area\ of\ circle}{area\ of\ square} = \frac{\pi \times r \times r}{(2\times r)^2} = \frac{\pi \times r \times r}{4 \times r \times r} = \frac{\pi}{4}$
Therefore, we get the formula to calculate π using Monte Carlo method:

$\pi = 4 \times \frac{number\ of\ points\ hitting\ circle\ area}{number\ of\ points\ hitting\ square\ area}$

In real world, we can only throw finite number of random points, therefore, the π calculated using the above formula is an approximation of the exact value of π.

We can have computers generate random numbers to simulate the throwing of points. For each point, we can have computer to generate two random floating point numbers to be the x and y coordinates of the point, where $0 \leq x \leq 2r$ and $0 \leq y \leq 2r$ so that (x,y) must fall within the square area. However, this randomly generated point could fall within the circle area or fall out of the circle area.

To decide if the randomly generated point (x,y) is within the area of the circle, we can compare the distance of the point to the center with the radius r. For example, the point A in the above figure is in the circle area since its

distance to center is less than r. However, the point B in the above figure is not in the circle area since its distance to center is greater than r.

Note: If the distance of point to the center equals to radius r, then that point is considered within the circle area.

Assume the radius of the circle is r and the coordinates of randomly generated point P is (x, y), then the distance of P to center is:
$Distance(P,Center) = \sqrt{(x-r)\times(x-r)+(y-r)\times(y-r)}$

To avoid the square root calculation, you can compare $Distance(P,Center) \times Distance(P,Center)$ with radius squared $r \times r$ in order to decide if the randomly generated point is within the circle area.

At the beginning, your program should ask for the inputs of radius r and the number of random points N the computer needs to generated. The output should like this:
Enter the radius of circle: 5
Enter the number of points: 10

During the generation, whenever a random point is generated, your program should print out the coordinates of the point and if the point is *In* or *Out* the circle area like this:

Point No.1(x=0.000000,y=6.551714): OUT
Point No.2(x=3.048189,y=6.749779): IN
Point No.3(x=1.067537,y=5.165868): IN
Point No.4(x=4.896695,y=6.024659): IN
Point No.5(x=3.699454,y=2.566607): IN
Point No.6(x=3.741874,y=8.255867): IN
Point No.7(x=1.727042,y=2.977996): IN
Point No.8(x=6.435438,y=7.896664): IN
Point No.9(x=9.878231,y=8.005921): OUT
Point No.10(x=4.642476,y=5.389874): IN

At the end, your program should output the number of points within and out of the circle, together with the approximation value of π like this:
*/******In Summary******/*
Points within circle area: 8
Points out of circle area: 2
Pi= 3.200000

To show that your program works correctly, run it once with the radius=10 and the number of points=20. Submit the output as your script file (mc.script). Please compile your C code using **-ansi -Wall** options as below to specify ANSI code with all warnings:

```
gcc -o mc -ansi -Wall mc.c
```

Please submit `mc.c`, `mc.script`, and `mc.txt` files. File `mc.txt` should briefly explain your program design within 10 sentences.

HINT In homework4, you have learned how to generate random integer numbers within the range of [0, n) (n is exclusive). However, in this homework, you need to generate floating point numbers with the range of [0, n] (n is inclusive). Therefore, there are some modification of the code described in homework4.

You need to replace the following line in homework4
int randomNumber = rand() % n; with
*double randomNumber = ((double)rand())/((double)RAND_MAX)*n;* /* n is the side of the square */

Furthermore, in howework5, we want you to use the same seed for the random numbers generation in order to generate the same series of random numbers. Therefore, get rid of the following line:
#include ⟨time.h⟩

and replace the following line in homework4
srand(time(NULL)); with
srand(0);

3 Bonus Problem [5 points]

Could we use Monte Carlo method to calculate Pi given the figure2 below? It's a quadrant of a circle circumscribed by a square. The radius of circle equals to the side of square and their centers are overlapped.

If yes, explain your method and formula in `mc_extra.txt` concisely within 10 sentences.

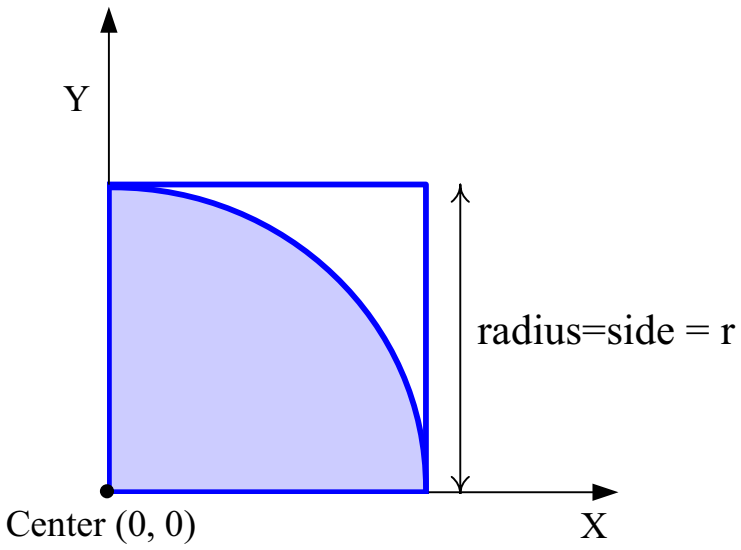

Figure 2: A Quadrant of a Circle Circumscribed by a Square

4 Submission

Submission for these files will be similar to last week's assignment. The only difference is that you need to create a directory called **`hw5/`**. Put all the files for assignment 5 in that directory and run the **`/ecelib/bin/turnin`** command to submit your homework.