IIT MADRAS

Indian Institute of Technology Madras

M.Tech Computer Science Information security.

Cryptography Basic.(CS6530)

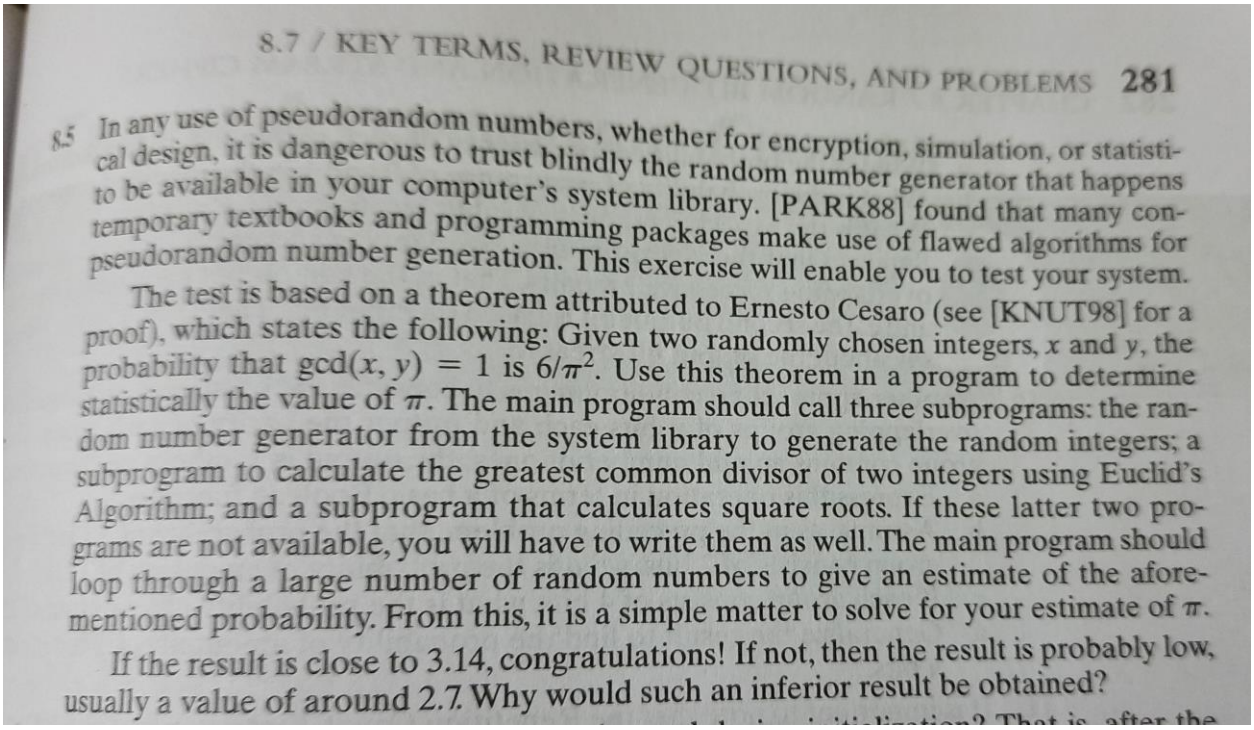
Assignment-2. (Question 8.5 Page number-281 prgramming)

Roll Number - CS21M515

Name – Ayub Shaikh.

Batch Year-2021

(Question:- 8.5 - Page number-281 prgramming)



8.7 / KEY TERMS, REVIEW QUESTIONS, AND PROBLEMS 281

8.5 In any use of pseudorandom numbers, whether for encryption, simulation, or statistical design, it is dangerous to trust blindly the random number generator that happens to be available in your computer's system library. [PARK88] found that many contemporary textbooks and programming packages make use of flawed algorithms for pseudorandom number generation. This exercise will enable you to test your system.

The test is based on a theorem attributed to Ernesto Cesaro (see [KNUT98] for a proof), which states the following: Given two randomly chosen integers, x and y, the probability that $\gcd(x, y) = 1$ is $6/\pi^2$. Use this theorem in a program to determine statistically the value of π. The main program should call three subprograms: the random number generator from the system library to generate the random integers; a subprogram to calculate the greatest common divisor of two integers using Euclid's Algorithm; and a subprogram that calculates square roots. If these latter two programs are not available, you will have to write them as well. The main program should loop through a large number of random numbers to give an estimate of the aforementioned probability. From this, it is a simple matter to solve for your estimate of π.

If the result is close to 3.14, congratulations! If not, then the result is probably low, usually a value of around 2.7. Why would such an inferior result be obtained?

Solution and Answer:-

I Have did the above example by using the C++ program.

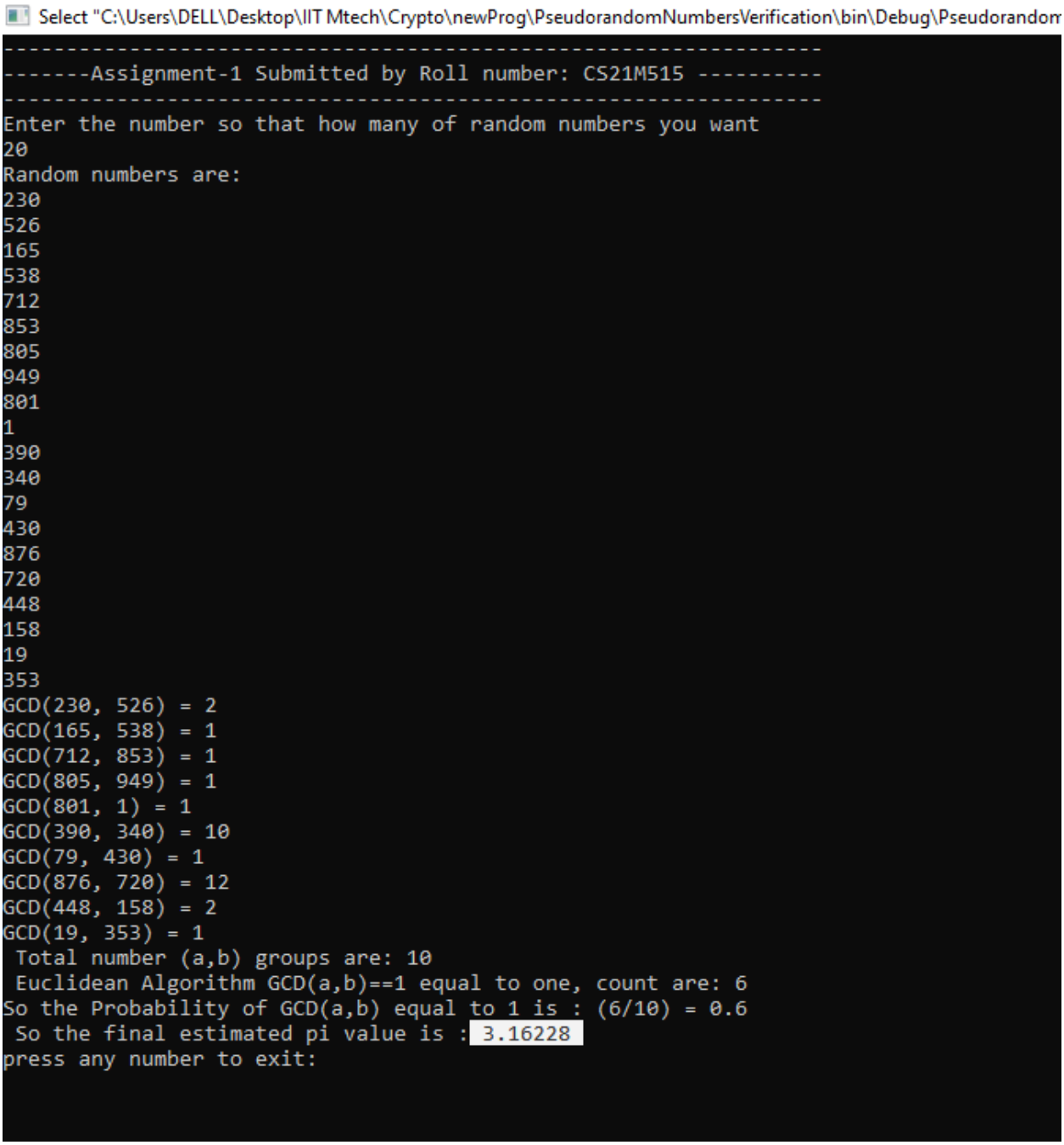
And .exe is shared in the google drive and also the program is shared.

Below are the screen shots of the C++ programm executed .

So in this belwo screen shot I have generated 20 random numbers and then calculated their GCD and propbaility of that and then value of Pi.

So in below screen shot as we can see that I am getting the approximate value of pi is = **3.16228**

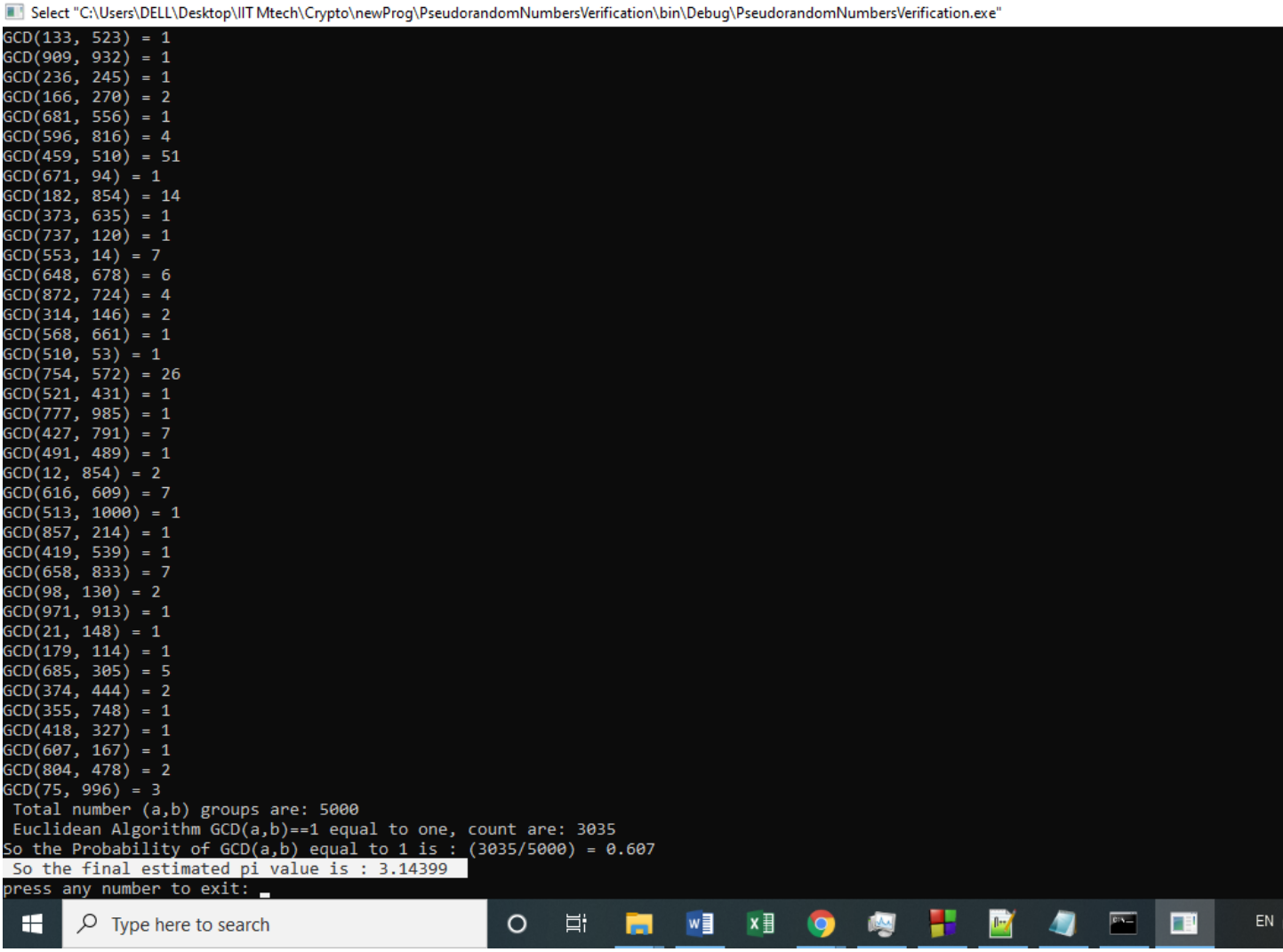
So it means the random numbers are generated properly and program is also correct.



So in this belwo screen shot I have generated 10000 random numbers and then calculated their GCD and propbaility of that and then value of Pi.

So in below screen shot as we can see that I am getting the approximate value of pi is = **3.14399**

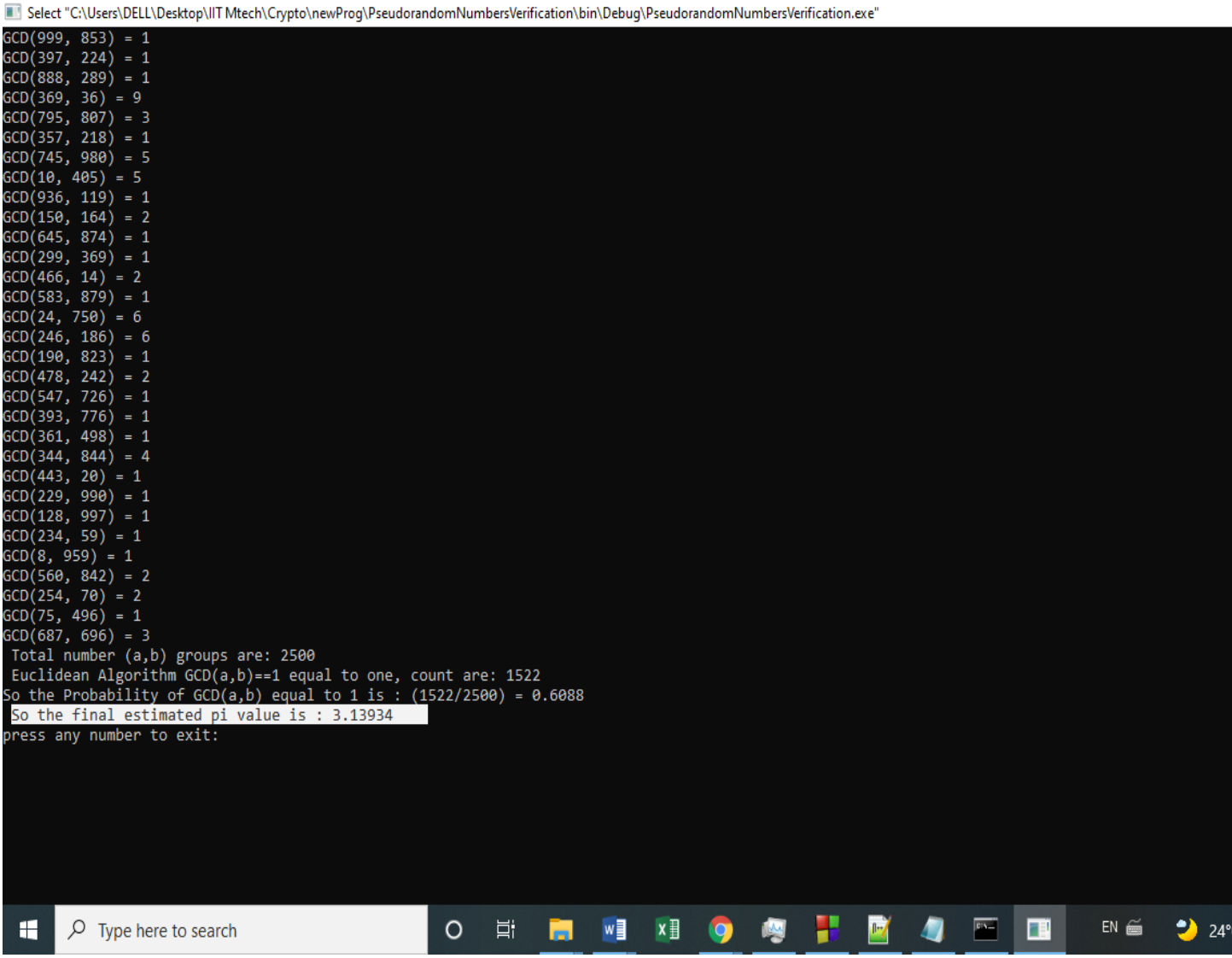
So it means the random numbers are generated properly and program is also correct.



So in this I have generated 5000 random numbers and then calculated their GCD and propbaility of that and then value of Pi.

So in below screen shot as we can see that I am getting the approximate value of pi is = **3.13934**

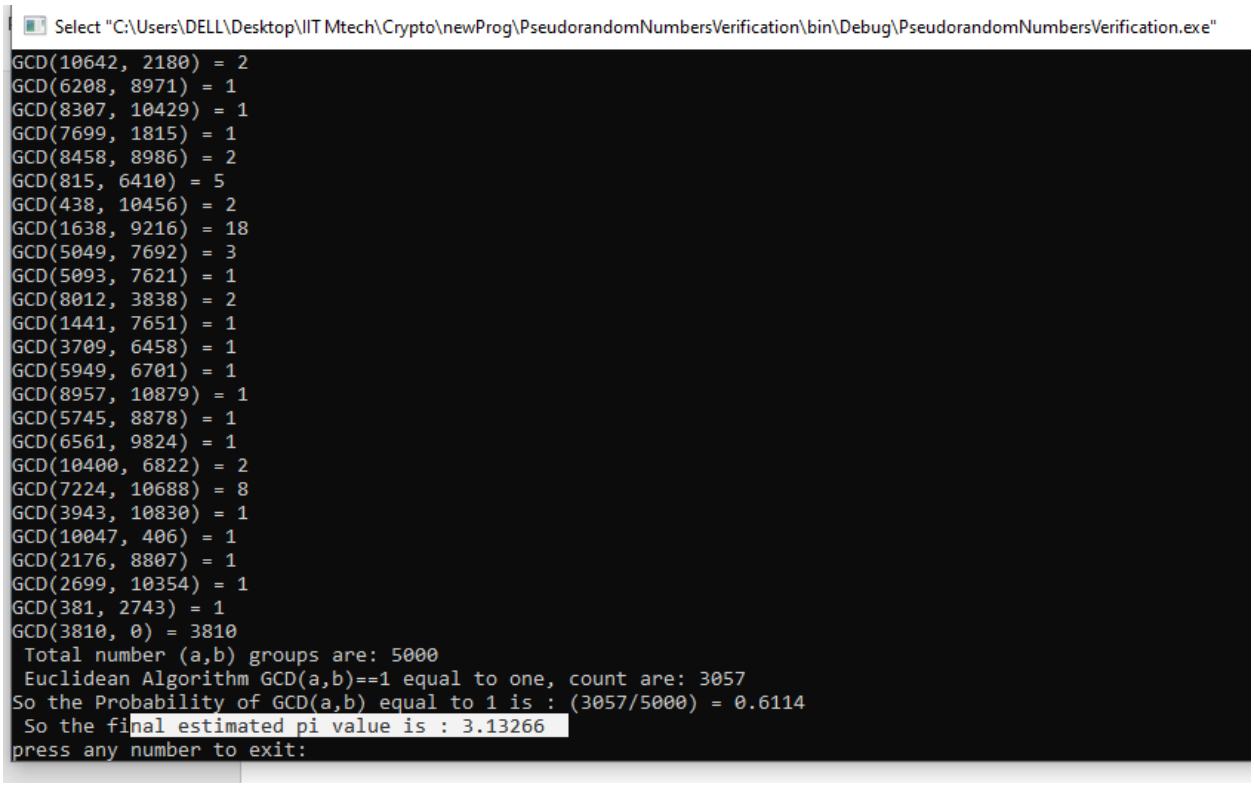
So it means the random numbers are generated properly and program is also correct.

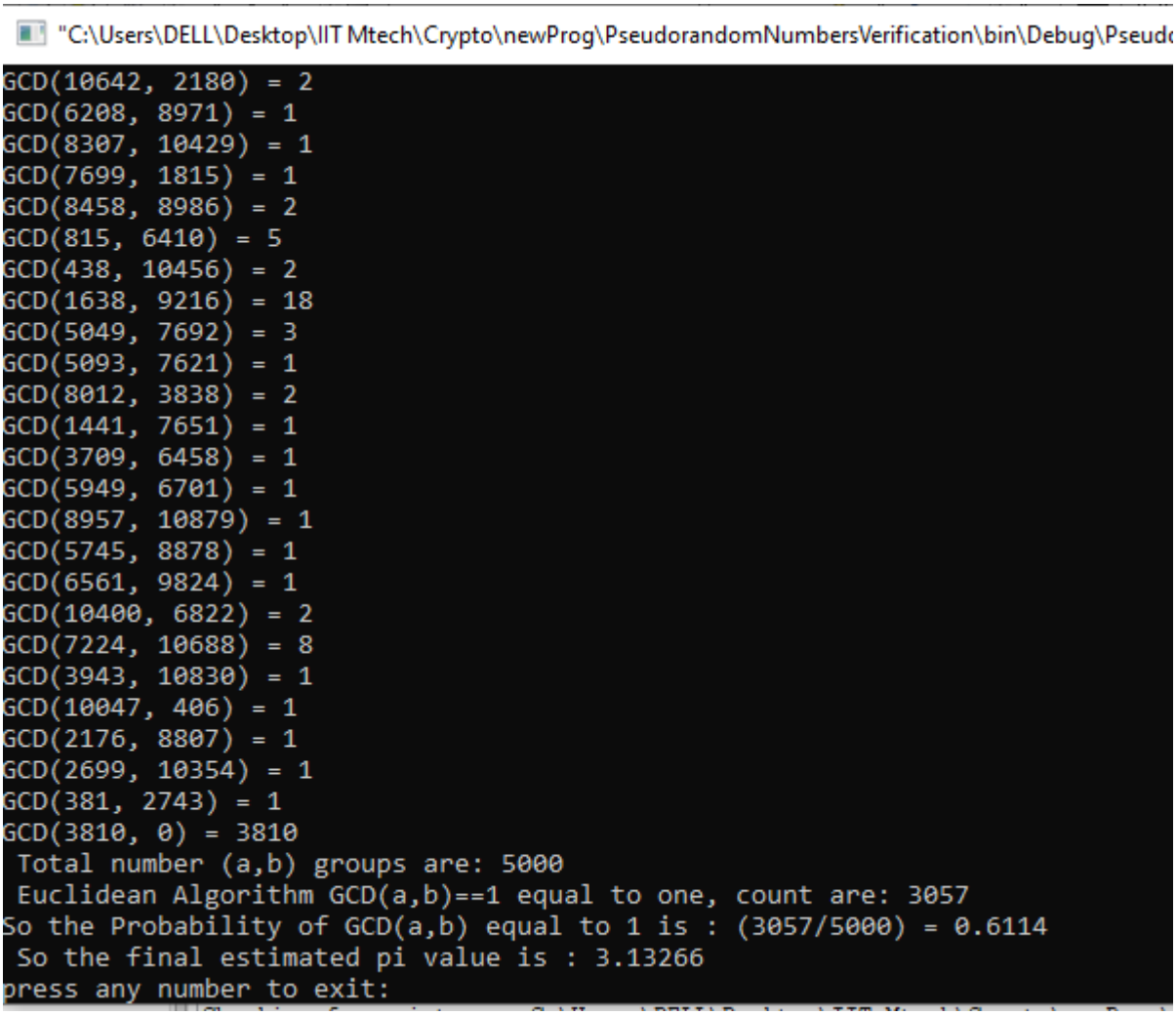


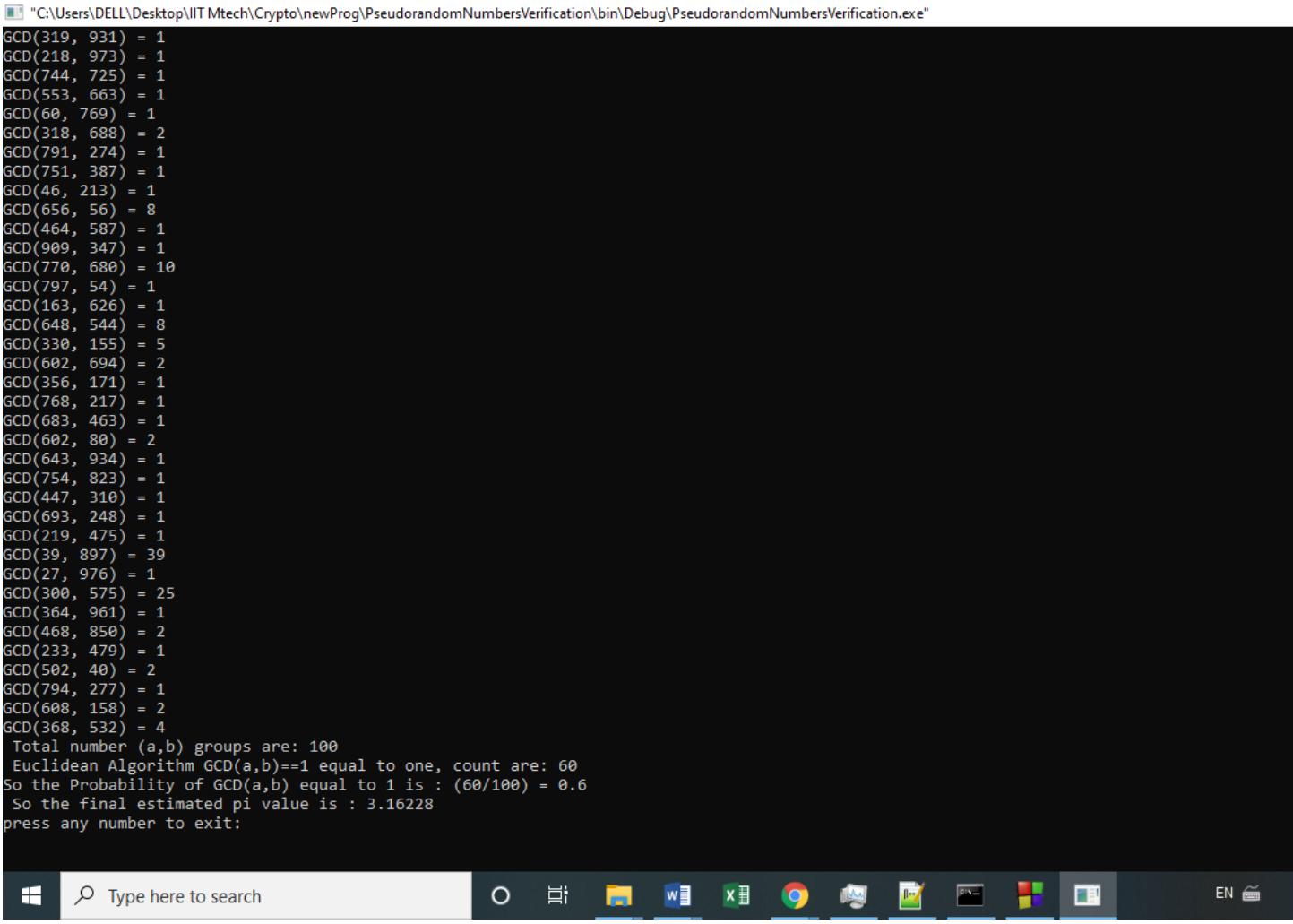
So in this I have generated 10000 random numbers and then calculated their GCD and propbaility of that and then value of Pi.

So in below screen shot as we can see that I am getting the approximate value of pi is = **3.13266**

So it means the random numbers are generated properly and program is also correct.


Select "C:\Users\DELL\Desktop\IIT Mtech\Crypto\newProg\PseudorandomNumbersVerification\bin\Debug\PseudorandomNumbersVerification.exe"
GCD(10642, 2180) = 2
GCD(6208, 8971) = 1
GCD(8307, 10429) = 1
GCD(7699, 1815) = 1
GCD(8458, 8986) = 2
GCD(815, 6410) = 5
GCD(438, 10456) = 2
GCD(1638, 9216) = 18
GCD(5049, 7692) = 3
GCD(5093, 7621) = 1
GCD(8012, 3838) = 2
GCD(1441, 7651) = 1
GCD(3709, 6458) = 1
GCD(5949, 6701) = 1
GCD(8957, 10879) = 1
GCD(5745, 8878) = 1
GCD(6561, 9824) = 1
GCD(10400, 6822) = 2
GCD(7224, 10688) = 8
GCD(3943, 10830) = 1
GCD(10047, 406) = 1
GCD(2176, 8807) = 1
GCD(2699, 10354) = 1
GCD(381, 2743) = 1
GCD(3810, 0) = 3810
Total number (a,b) groups are: 5000
Euclidean Algorithm GCD(a,b)==1 equal to one, count are: 3057
So the Probability of GCD(a,b) equal to 1 is : (3057/5000) = 0.6114
So the final estimated pi value is : 3.13266
press any number to exit:


"C:\Users\DELL\Desktop\IIT Mtech\Crypto\newProg\PseudorandomNumbersVerification\bin\Debug\Pseudo
GCD(10642, 2180) = 2
GCD(6208, 8971) = 1
GCD(8307, 10429) = 1
GCD(7699, 1815) = 1
GCD(8458, 8986) = 2
GCD(815, 6410) = 5
GCD(438, 10456) = 2
GCD(1638, 9216) = 18
GCD(5049, 7692) = 3
GCD(5093, 7621) = 1
GCD(8012, 3838) = 2
GCD(1441, 7651) = 1
GCD(3709, 6458) = 1
GCD(5949, 6701) = 1
GCD(8957, 10879) = 1
GCD(5745, 8878) = 1
GCD(6561, 9824) = 1
GCD(10400, 6822) = 2
GCD(7224, 10688) = 8
GCD(3943, 10830) = 1
GCD(10047, 406) = 1
GCD(2176, 8807) = 1
GCD(2699, 10354) = 1
GCD(381, 2743) = 1
GCD(3810, 0) = 3810
Total number (a,b) groups are: 5000
Euclidean Algorithm GCD(a,b)==1 equal to one, count are: 3057
So the Probability of GCD(a,b) equal to 1 is : (3057/5000) = 0.6114
So the final estimated pi value is : 3.13266
press any number to exit:



Program in C++ language

```
#include<iostream>

#include<cstdlib>

#include <ctime>

#include <cmath>


//-------Assignment-2 Submitted by Roll number: CS21M515 -

using namespace std;
```

```
 // GCD

// Function to return
// gcd of a and b
int gcd(int a, int b)
{
   if (a == 0)
      return b;
   return gcd(b % a, a);
}
   /// end


int main(){

   int n;
   int m;
   int t;
   int c;

   cout << "-----------------------------------------------------------------" << endl;
   cout << "-------Assignment-2 Submitted by Roll number: CS21M515 ----------" << endl;
   cout << "-----------------------------------------------------------------" << endl;
   cout << "Enter the number so that how many of random numbers you want" << endl;
   cin >> n;

  cout << "Random numbers are:" << endl;
```

```
 int randomNumArray[n];
  float prob;
int range = 1000;

if(n > range){
   range = n + range;
}
   // Providing a seed value
   srand((unsigned) time(NULL));
 m = n+1;
   // Loop to get 5 random numbers

   for(int i=0; i < n; i++){

      // Retrieve a random number between 100 and 200
      // Offset = 1
      // Range = 1000

      int random = 1 + (rand() % range);

      randomNumArray[i] = random;
      // Print the random number
      cout<<random<<endl;
   }
float euclideanGCDCount = 0.0;
```

```
    for(int i=0; i < n; i++){
      // cout << randomNumArray[i] << "  " << endl;

       int a = randomNumArray[i];

       int b = randomNumArray[++i];

       int g = gcd(a, b);
       cout << "GCD(" << a << ", " << b << ") = " << g << endl;

       if(g == 1){
         euclideanGCDCount++;

       }

    }

//     cout << "i(" << i  << endl;

           float numGroups = n/2;
        cout << " Total number (a,b) groups are: " << numGroups << endl;
       cout << " Euclidean Algorithm GCD(a,b)==1 equal to one, count are: " <<
euclideanGCDCount << endl;

        prob = euclideanGCDCount / numGroups;

         cout << "So the Probability of GCD(a,b) equal to 1 is : (" << euclideanGCDCount << "/"
<< numGroups << ") = " << prob << endl;
```

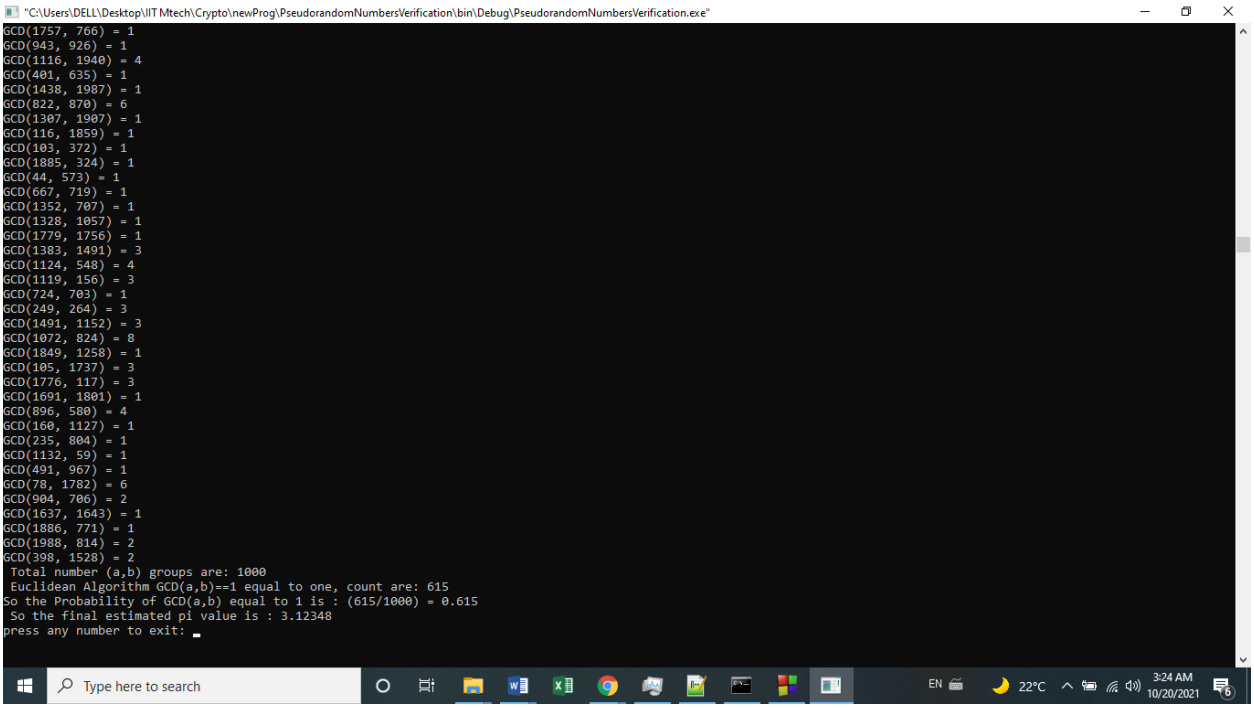
```
    float cal = 6/prob;
    float pi = sqrt(cal);


     cout << " So the final estimated pi value is : " << pi << endl;

   int ex;
   cout << "press any number to exit: ";
   cin >> ex;
   if(ex>0){
     return 0;
   }

  //return 1;

}
```



THANK You Sir .... ☺☺☺