SIDDAGANGA INSTITUTE OF TECHNOLOGY, TUMKUR-572103

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CRYPTOGRAPHY AND NETWORK SECURITY LAB (7RCSL01)

Student Name: Pragati Shankar	**USN: 1SI19CS090**	**Batch No: B2**	**Date: 20-12-2022**
Evaluation:			

Write Up (10 marks)	**Clarity in concepts (10 marks)**	**Implementation and execution of the algorithms (10 marks)**	**Viva (05 marks)**	**Total (35 marks)**

Sl.No	**Name of the Faculty In-Charge**	**Signature**
1.	H K Vedamurthy	
2.	Gururaj S P	

Question No: 10

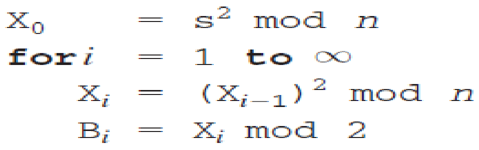
Write a program to generate large random number using BBS random number generator algorithm and check whether the generated number is prime or not using RABIN-MILLER Primality testing algorithm.

Algorithm:

BBS Random Number Generator Algorithm:

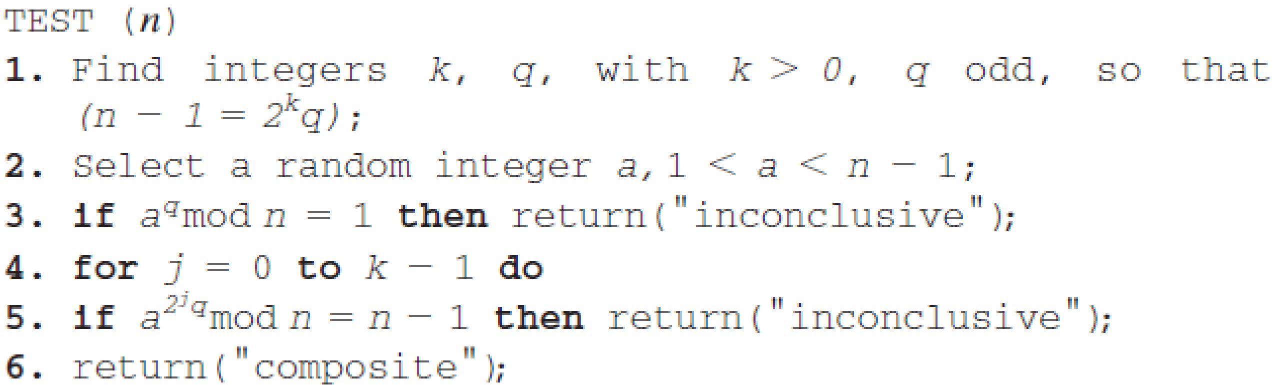
First, choose two large prime numbers p and q, that both have a remainder of 3 when divided by 4.

P=Q=3 mod 4



```
$X_0 = s^2 \bmod n$
for $i = 1$ to $\infty$
    $X_i = (X_{i-1})^2 \bmod n$
    $B_i = X_i \bmod 2$
```

RABIN-MILLER Primality testing algorithm:



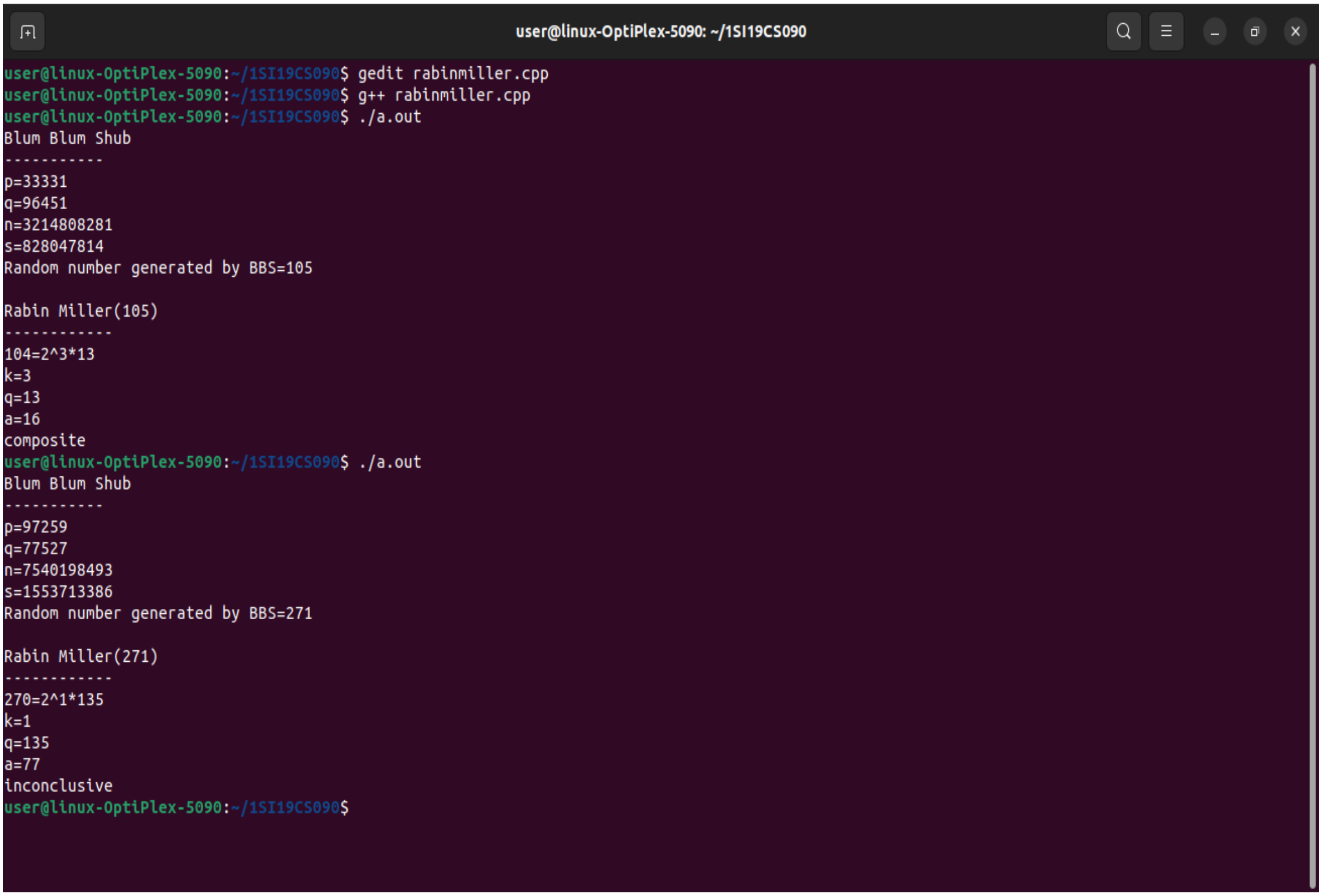
```
TEST ($n$)
1. Find integers $k$, $q$, with $k > 0$, $q$ odd, so that
   ($n - 1 = 2^k q$);
2. Select a random integer $a$, $1 < a < n - 1$;
3. if $a^q \bmod n = 1$ then return("inconclusive");
4. for $j = 0$ to $k - 1$ do
5. if $a^{2^j q} \bmod n = n - 1$ then return("inconclusive");
6. return("composite");
```

Code:

```
#include <bits/stdc++.h>
using namespace std;

int randInRange(int low, int high)
{
    return rand()%(high-(low+1))+(low+1);
}

int genPrime3mod4()
{
    while(true)
    {
        int num=randInRange(10000,100000);
        if(num%4 !=3) continue;
        bool prime=true;
        for(int i=2;i<=sqrt(num);i++)
        {
            if(num% i ==0)
            {
                prime=false;
                break;
            }
        }
        if(prime) return num;
    }
}
int bbs(int p, int q)
{
    long long n=(long long)p*q;
    long long s;
    do
    {
        s=rand();
    }while(s%p==0 || s%q==0 || s==0);
    int B=0;
    long long x= (s*s)%n;
    for(int i=0;i<10;i++)
    {
        x=(x*x)%n;
        B=B<<1 | (x&1);
    }
    cout<<"Blum Blum Shub"<<endl<<"-----------"<<endl;
    cout<<"p="<<p<<"\nq="<<q<<"\nn="<<n<<"\ns="<<s<<endl;
    return B;
}
```

```
int powModN(int a,int b, int n)
{
        int res=1;
        for(int i=0;i<b;i++)
        {
                res=(res*a)%n;
        }
        return res;
}
string rabinMiller(int n)
{
        int k=0;
        int q=n-1;
        while(q%2==0)
        {
                q=q/2;
                k++;
        }
        int a=randInRange(1,n-1);
        cout<<"\nRabin Miller("<<n<<")\n------------"<<endl;
        cout<<n-1<<"=2^"<<k<<"*"<<q<<endl;
        cout<<"k="<<k<<"\nq="<<q<<"\na="<<a<<endl;

        if(powModN(a,q,n)==1) return "inconclusive";

        for(int j=0;j<k;j++)
        {
                if(powModN(a,pow(2,j)*q,n)==n-1) return "inconclusive";
        }
        return "composite";
}
int main()
{
        srand(time(NULL));
        int p=genPrime3mod4();
        int q=genPrime3mod4();
        int randNum=bbs(p,q);
        cout<<"Random number generated by BBS="<<randNum<<endl;
        cout<<rabinMiller(randNum)<<endl;
}
```

Output Screenshot:



```
user@linux-OptiPlex-5090:~/1SI19CS090$ gedit rabinmiller.cpp
user@linux-OptiPlex-5090:~/1SI19CS090$ g++ rabinmiller.cpp
user@linux-OptiPlex-5090:~/1SI19CS090$ ./a.out
Blum Blum Shub
-----------
p=33331
q=96451
n=3214808281
s=828047814
Random number generated by BBS=105

Rabin Miller(105)
------------
104=2^3*13
k=3
q=13
a=16
composite
user@linux-OptiPlex-5090:~/1SI19CS090$ ./a.out
Blum Blum Shub
-----------
p=97259
q=77527
n=7540198493
s=1553713386
Random number generated by BBS=271

Rabin Miller(271)
------------
270=2^1*135
k=1
q=135
a=77
inconclusive
user@linux-OptiPlex-5090:~/1SI19CS090$
```