Name: Sourav Bikash

Roll No.:11012338

Submission Date: 01/03/2013 Time:23:59 hrs.

Aim of the Problem:

The problem involves the use of the Box-Muller Method to generate standard normal numbers. The second part involves the use of a modification of the Box-Muller Method done by Marsaglia-Bray.

Mathematical Analysis/Theory:

Firstly we make use of the transformation theorem:

Theorem:

Suppose X is a random variable in IRn with density $f(x) > 0$ on the support S. The transformation $h : S \to B$, S,B IR^n is assumed to be invertible and the inverse be continuously differentiable on B. $Y := h(X)$ is the transformed random variable. Then Y has the density

$$f(h^{-1}(y)) = |\partial(x_1, \ldots, x_n)/\partial(y_1, \ldots, y_n)| \qquad , y \text{} B$$

where $x = h^{-1}(y)$ and $\partial(x_1, \ldots, x_n)/\partial(y_1, \ldots, y_n)$ is the determinant of the Jacobian matrix of all first-order derivatives of $h^{-1}(y)$.

In this assignment we make use of the transformation in $\mathbf{R}^2$ to generate normal variates.

Method of Box-Muller:

To apply the Theorem we start with the unit square S := $[0, 1]^2$ and the density of the bivariate uniform distribution. The transformation is

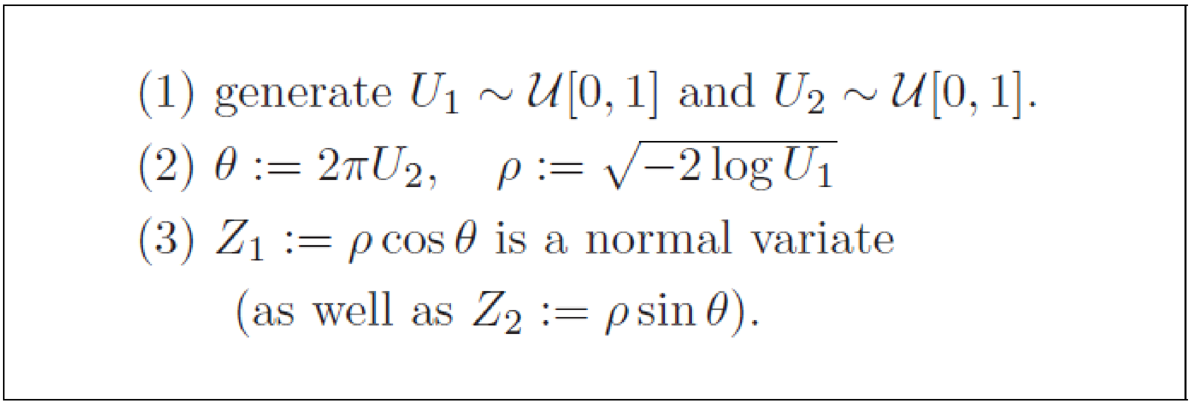
$$y_1 = \sqrt{(-2 \log x_1)}(\cos 2\pi x_2) =: h_1(x_1, x_2)$$
$$y_2 = \sqrt{(-2 \log x_1)}(\sin 2\pi x_2) =: h_2(x_1, x_2)$$

The function h(x) is defined on $[0, 1]^2$ with values in IR^2. The inverse function
h^{-1} is given by

$$x1 = \exp\{-(y^2{}_1 + y^2{}_2)/2\}$$

$$x2 = (\arctan(y_2/y_1))/2\pi$$

Therefore the Algorithm:



(1) generate $U_1 \sim \mathcal{U}[0,1]$ and $U_2 \sim \mathcal{U}[0,1]$.

(2) $\theta := 2\pi U_2, \quad \rho := \sqrt{-2\log U_1}$

(3) $Z_1 := \rho\cos\theta$ is a normal variate

(as well as $Z_2 := \rho\sin\theta$).

The Variant of Marsaglia:

The variant of Marsaglia prepares the input in the above Algorithm such that trigonometric functions are avoided. For $U \sim U[0, 1]$ we have $V := 2U- 1 \sim U[-1, 1]$. (Temporarily we misuse also the financial variable V for local purposes.) Two values V_1, V_2 calculated in this way define a point in the (V_1, V_2)-plane. Only points within the unit disk are accepted:

$D := \{(V_1, V_2) : V_1^2 + V_2^2 < 1\}$; accept only (V_1, V_2) ⍰ D .

In case of rejectance both values V_1, V_2 must be rejected. As a result, the surviving (V_1, V_2) are uniformly distributed on D with density $f(V_1, V_2) = 1/\pi$ for $(V_1, V_2) \in D$. A transformation from the disk D into the unit square $S := [0, 1]^2$ is defined by

$$x_1=V_1^2+V_2^2$$

$$x_2=(1/2\pi)(\arg(V_1,V_2))$$

That is the Cartesian Co-ordinates V_1,V_2 on D are mapped to the squared radius and the normalized angle. For illustration, see Figure. These “polar -coordinates” (x_1, x_2) are uniformly distributed on *S.*
With these variables the relations:

$$\text{Cos } 2\pi x_2=V_1/\sqrt{(V_1^2+V_2^2)}$$

$$\text{Sin } 2\pi x_2= V_2/\sqrt{(V_1^2+V_2^2)}$$

Hold, which means that it is no longer necessary to evaluate trigonometric relations in the Box-Muller Method.

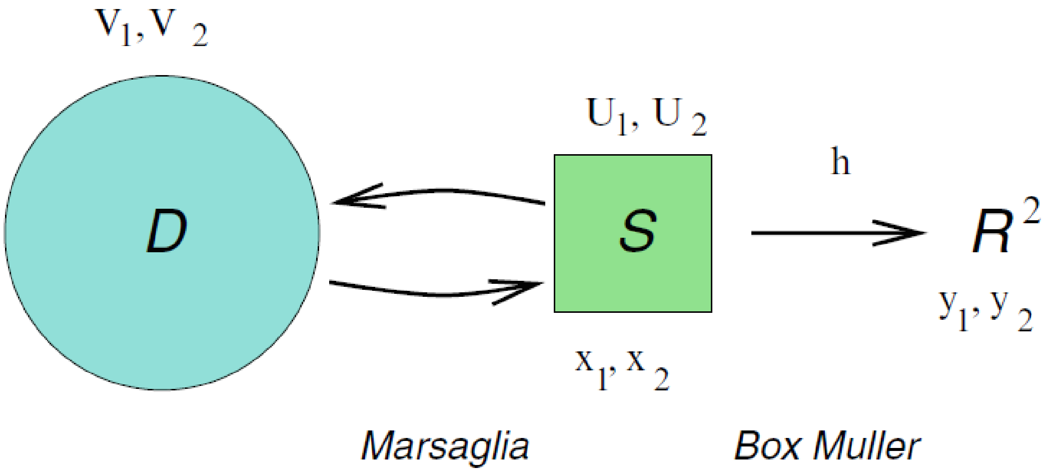
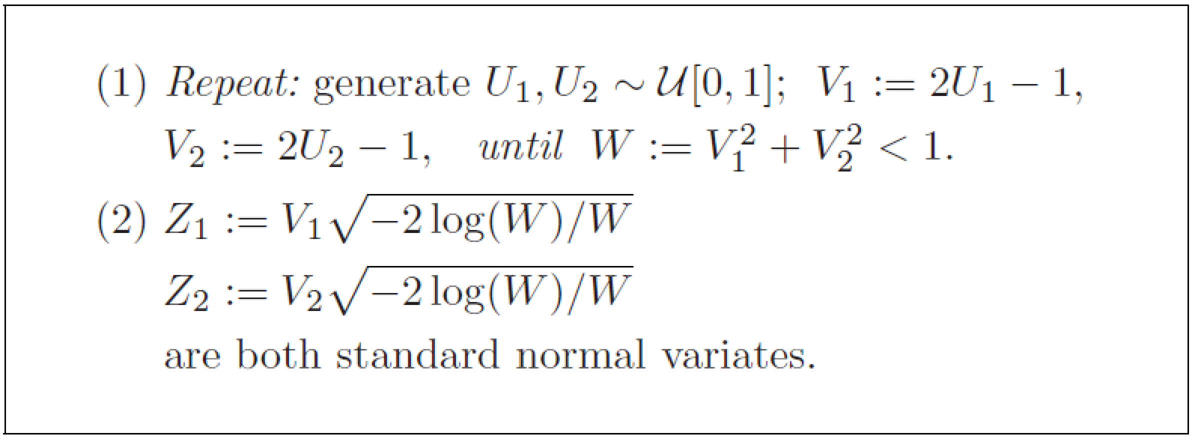


Figure describing the transformations of Box-Muller and Marsaglia Variant.

The Algorithm:



```
(1) Repeat: generate $U_1, U_2 \sim \mathcal{U}[0,1]$;  $V_1 := 2U_1 - 1$,
    $V_2 := 2U_2 - 1$,  until  $W := V_1^2 + V_2^2 < 1$.
(2) $Z_1 := V_1\sqrt{-2\log(W)/W}$
    $Z_2 := V_2\sqrt{-2\log(W)/W}$
    are both standard normal variates.
```

Part I:

This question wants to use the Box-Muller method and Marsaglia-Bray method to do the following :

(a) Generate a sample of 100, 500 and 10000 values from N (0, 1). Hence find the sample mean and variance.

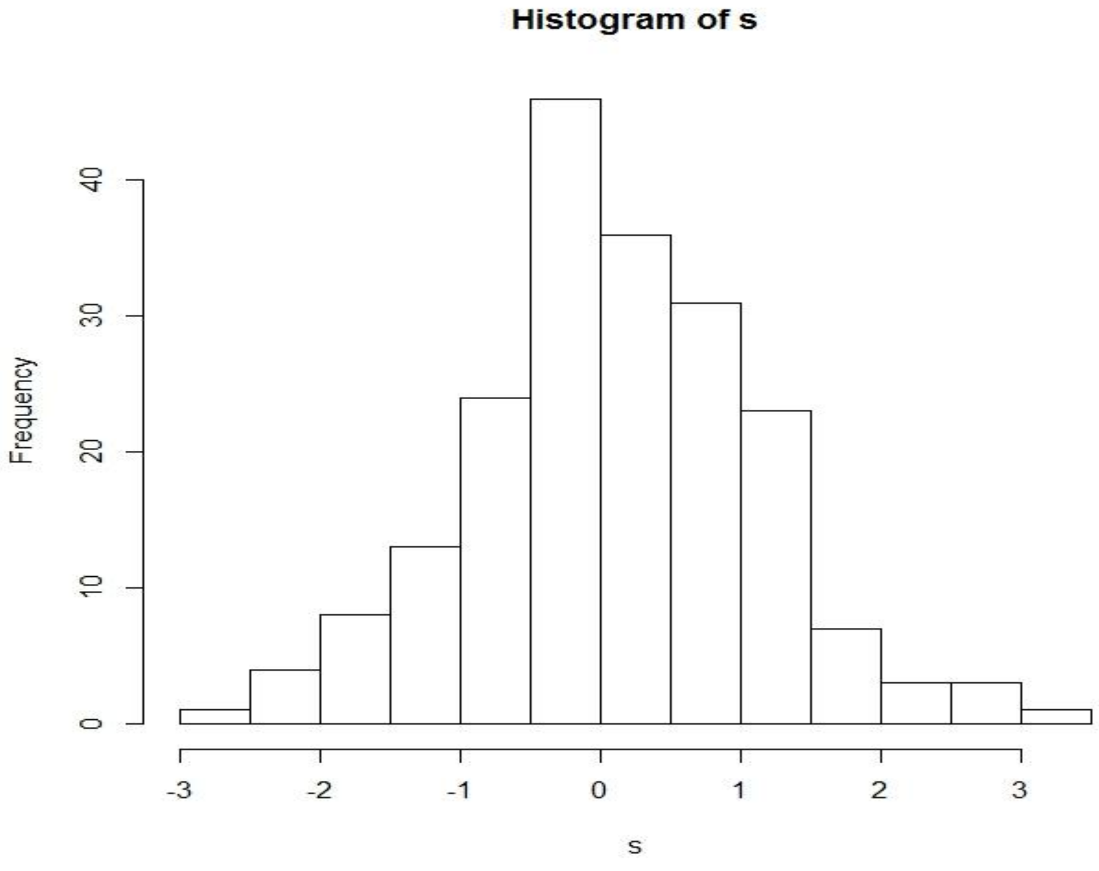
(b) Draw histogram in all cases.

Implementation using R (using the normal Box-Muller Method):

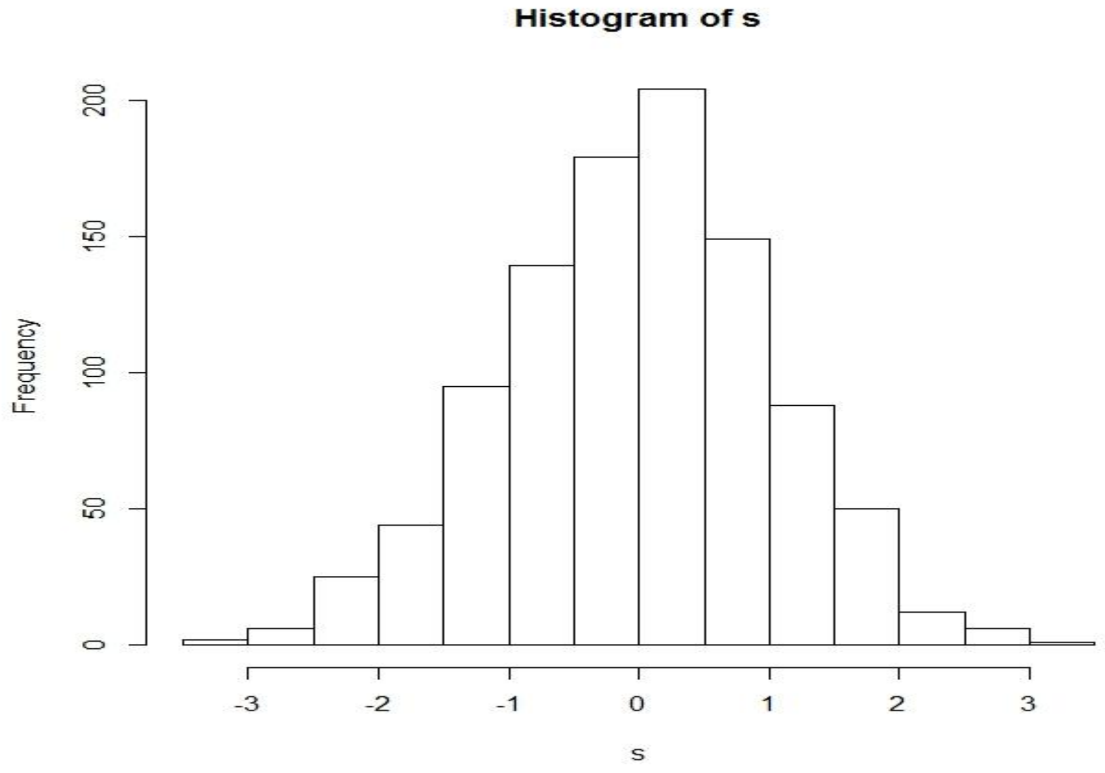
```
boxmuller<-function(n)

{

        RN<-NULL;

        for(i in 1:n)

        {
                u1<-runif(1,min=0,max=1);
                u2<-runif(1,min=0,max=1);
                r<--2*log(u1);
                v<-2*(3.14)*u2;
                z1<-(sqrt(r))*cos(v);
                z2<-(sqrt(r))*sin(v);
                RN<-c(RN,z1,z2);
        }
        return(RN);
}
s<-boxmuller(10000);
print(s);
```

<u>Using the output the following histograms were generated:</u>

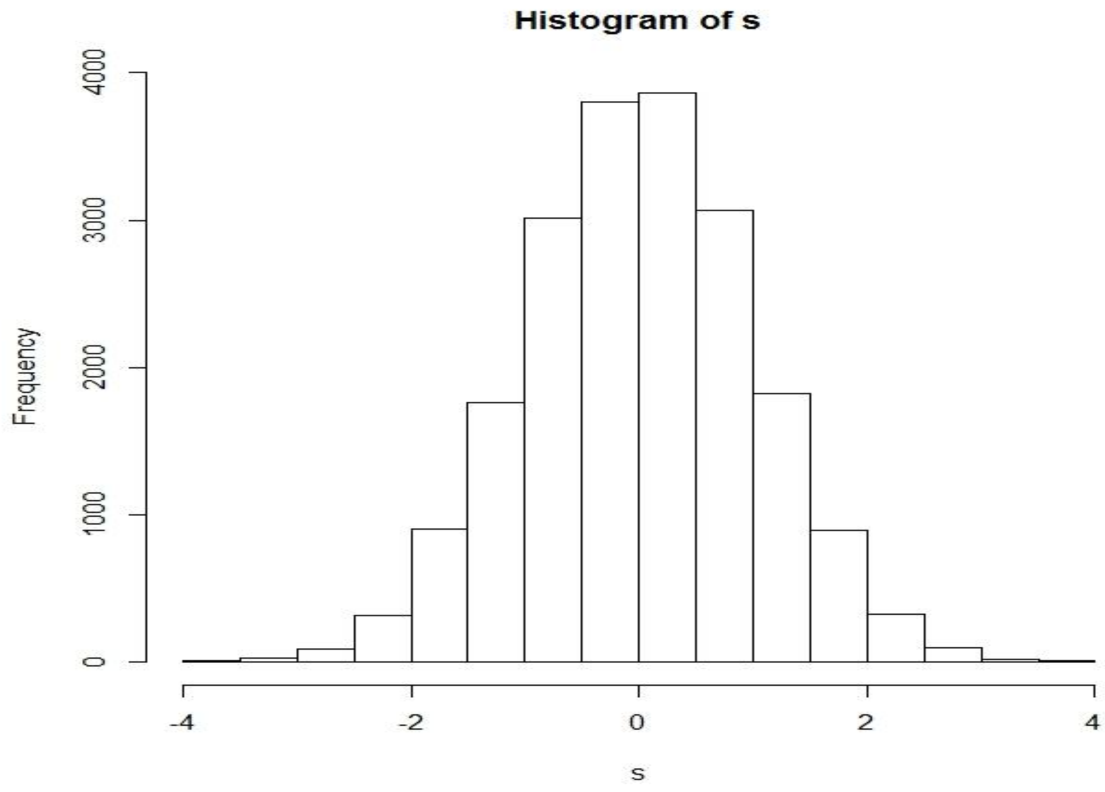
a) For N=100



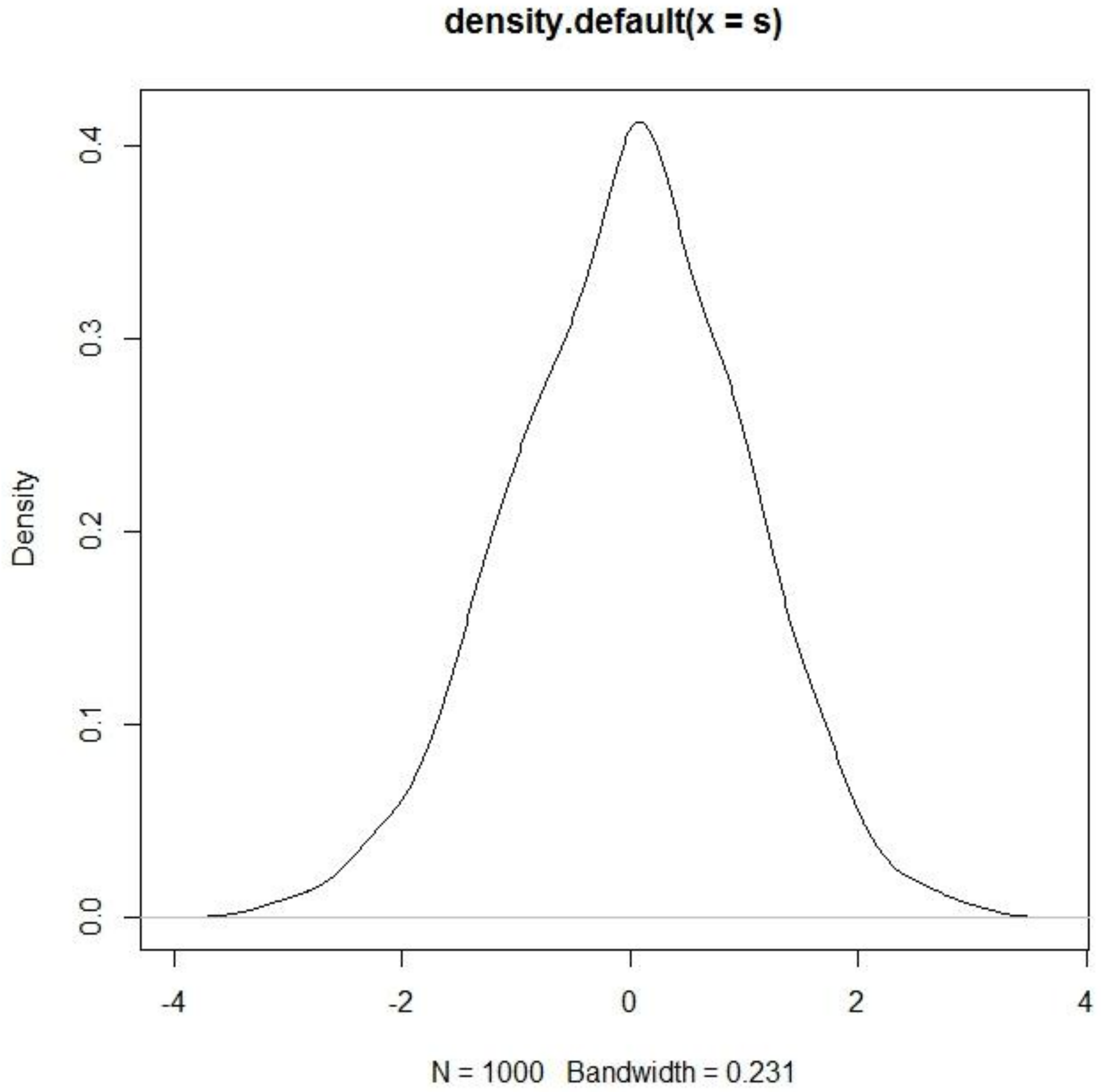
b) For N=500



c) For N=10000



The density plot of the so generated random numbers:



Conclusion:

By looking at the density plot of the numbers and comparing it with the standard normal distribution, we can say that our result is correct.

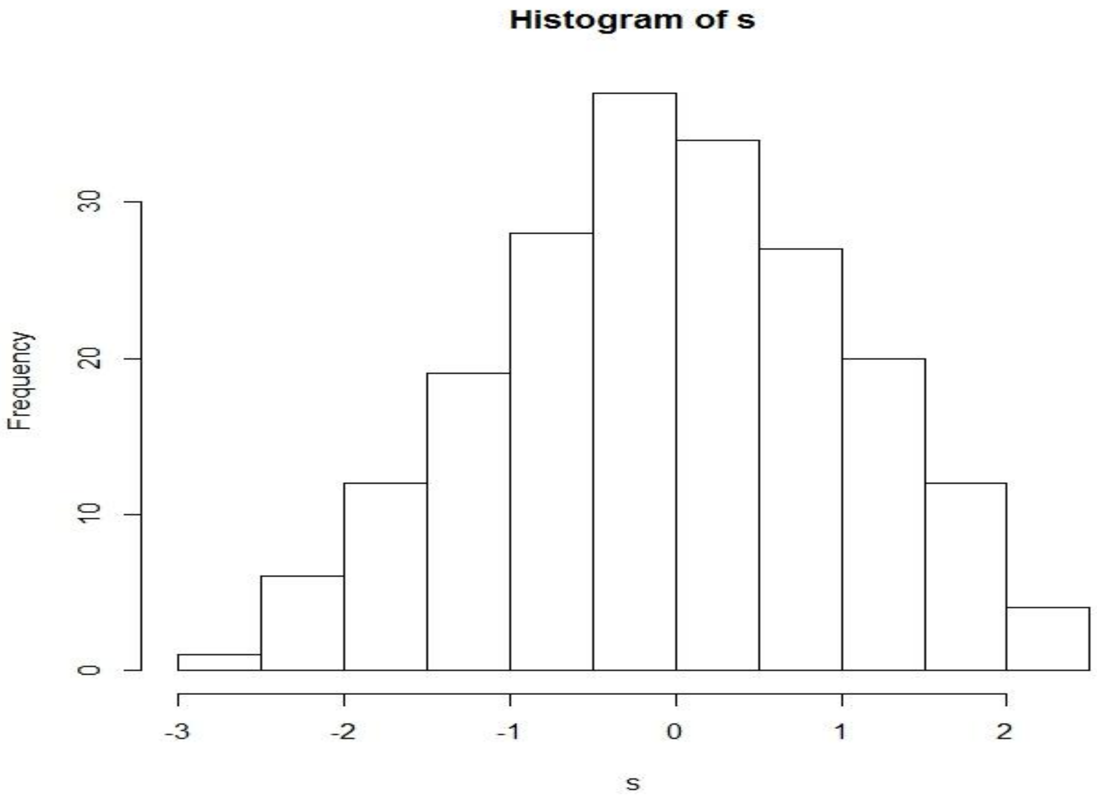
Implementation using R (using the Marsaglia Variant):-

```
#this is the r implementation of the box-muller method with marsaglia-bray modifications

boxmuller<-function(n)
{
        RN<-NULL;
        for(i in 1:n)
```

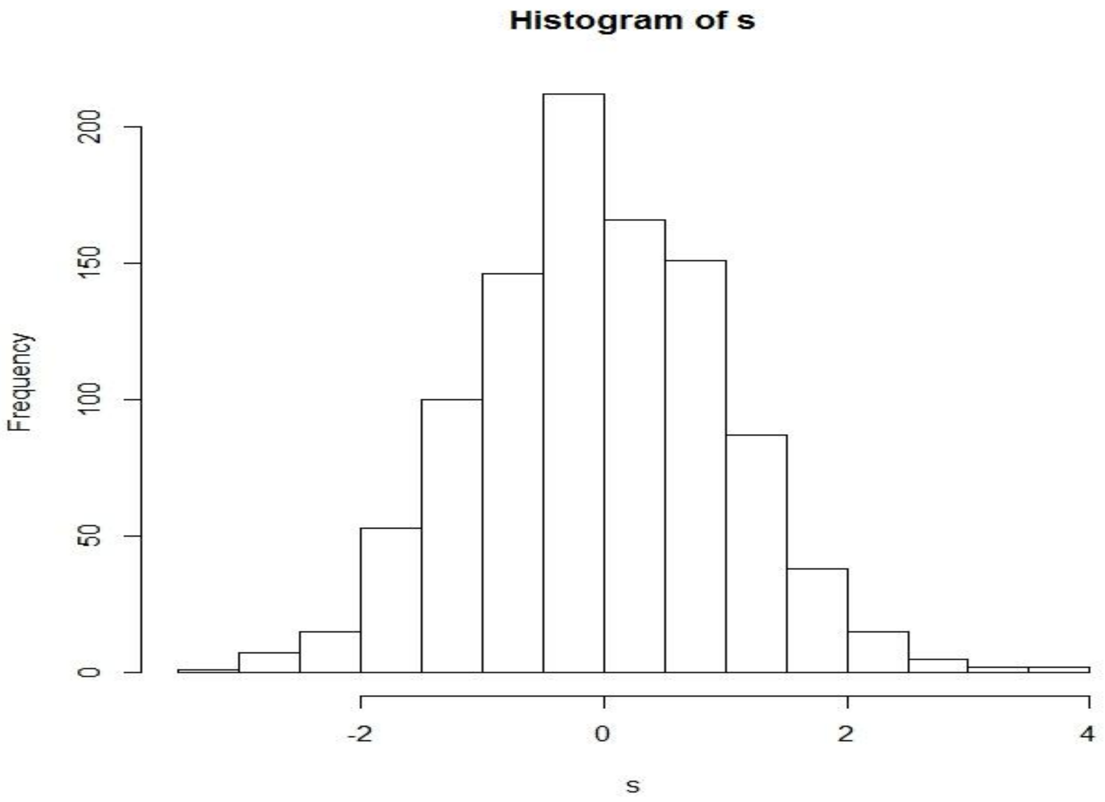
```
    {
        u1<-0;
        u2<-0;
        k<-2;
        while(k>1)
        {
            u1<-runif(1,min=0,max=1);
            u1<-(2*u1)-1;
            u2<-runif(1,min=0,max=1);
            u2<-(2*u2)-1;
            k<-(u1*u1)+(u2*u2);
        }
        #print(paste(k));
        r<-(-2)*log(k);
        cv<-u1/sqrt(k);
        sv<-u2/sqrt(k);
        z1<-sqrt(r)*cv;
        z2<-sqrt(r)*sv;
        RN<-c(RN,z1,z2);
    }
    return(RN);
}
s<-boxmuller(10000);

print(s);
```

<u>Using the output the following histograms were generated:</u>

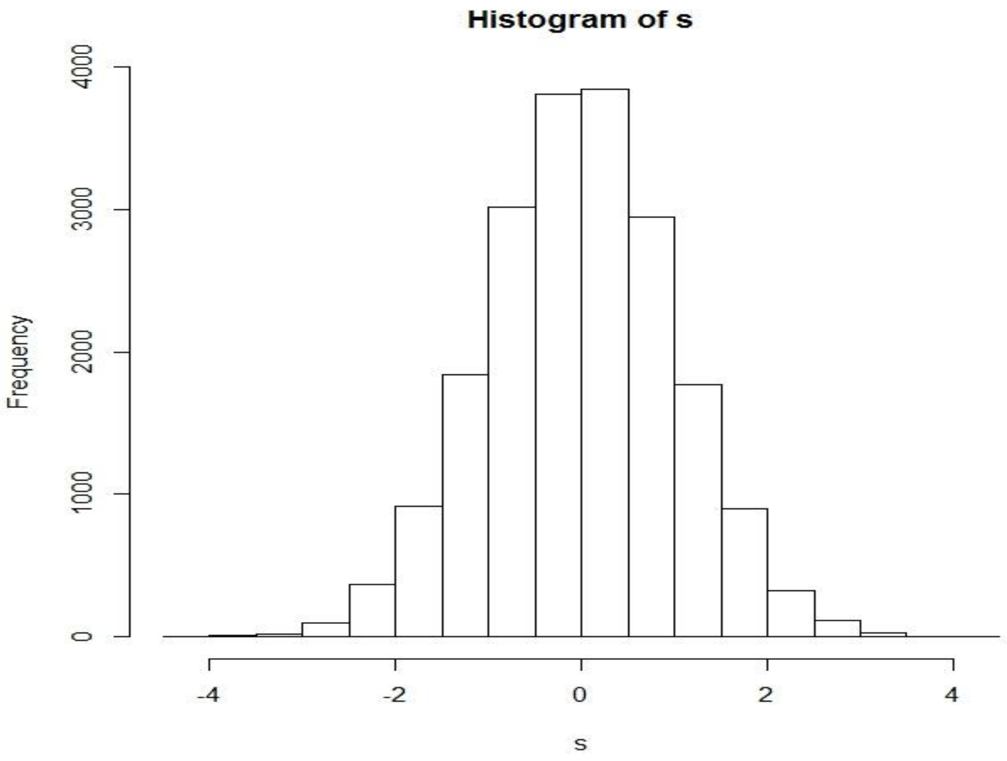
a) For N=100



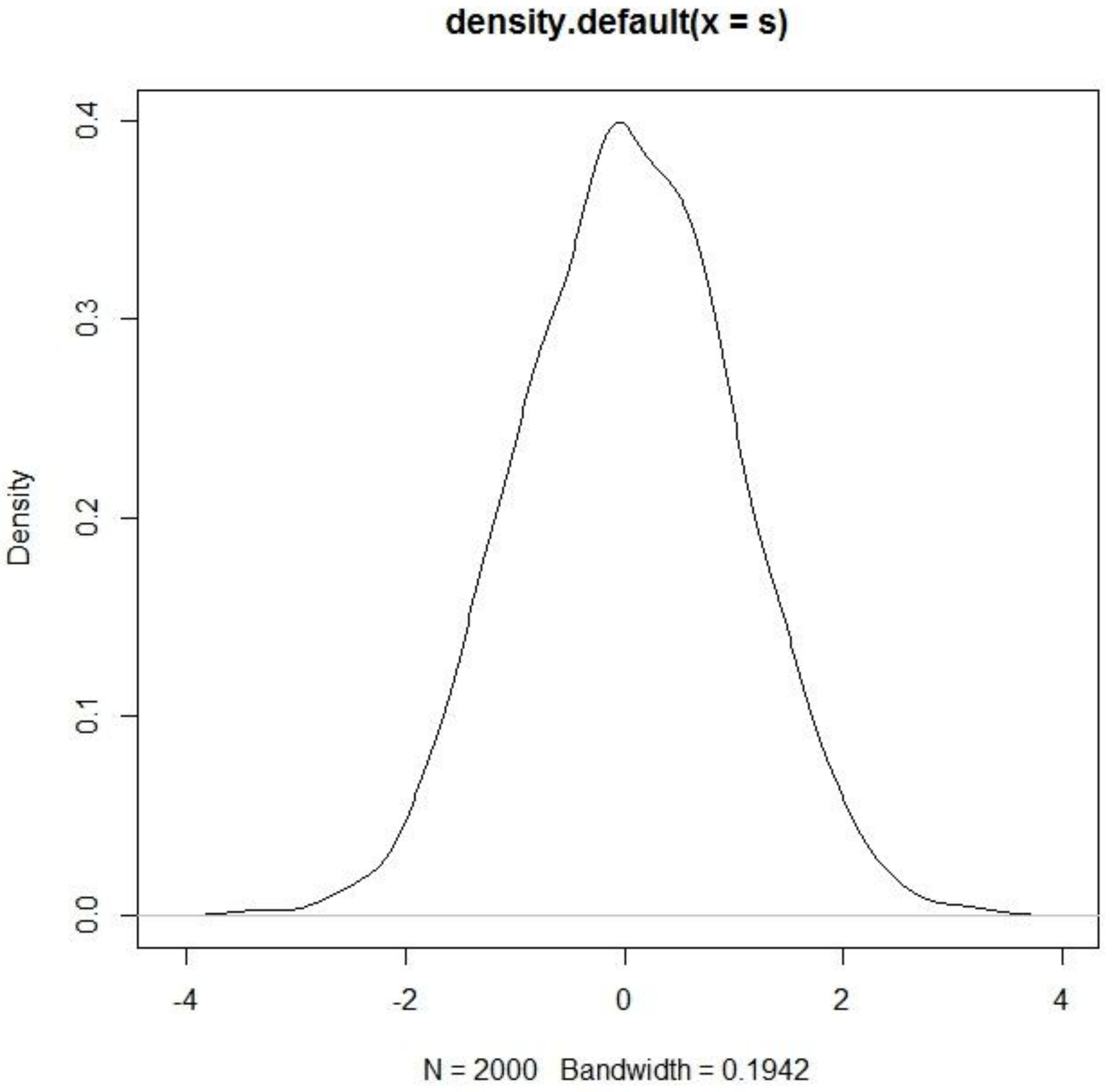
b)For N=500



c) For N=10000



The density plot of the generated numbers comes out to be like this:



Conclusion:

By looking at the density plot of the numbers and comparing it with the standard normal distribution, we can say that our result is correct.

Results:

The sample mean:

-0.01023877

The Sample Variance:

1.013356

Part II:

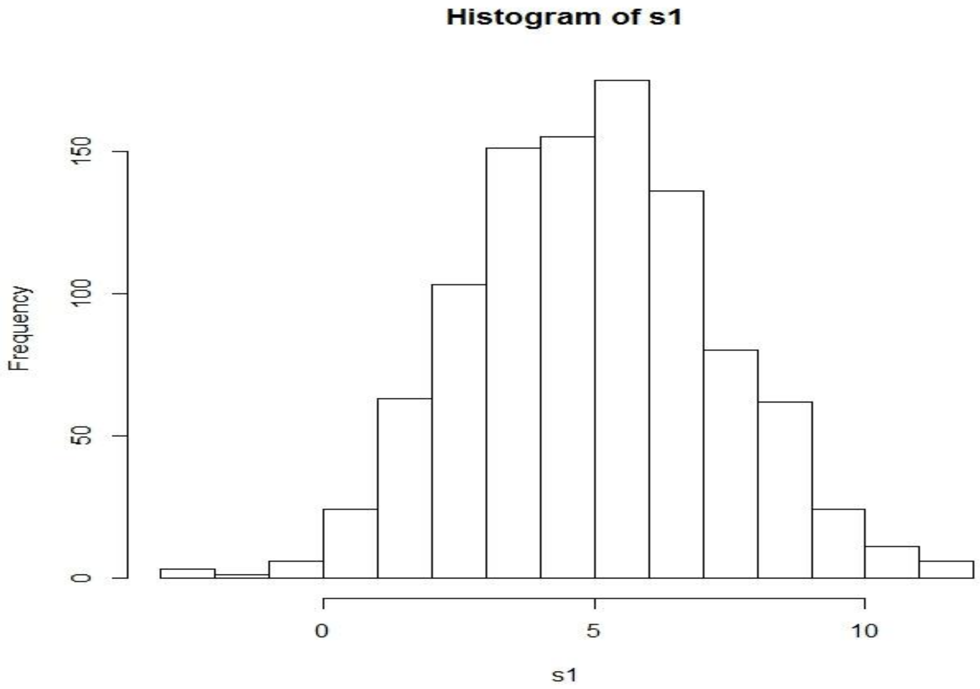
This question asks us to use the above generated values to generated samples from N(μ = 0, ϭ = 5) and N(μ = 5, ϭ = 5).

For N(μ = 5, ϭ = 5):

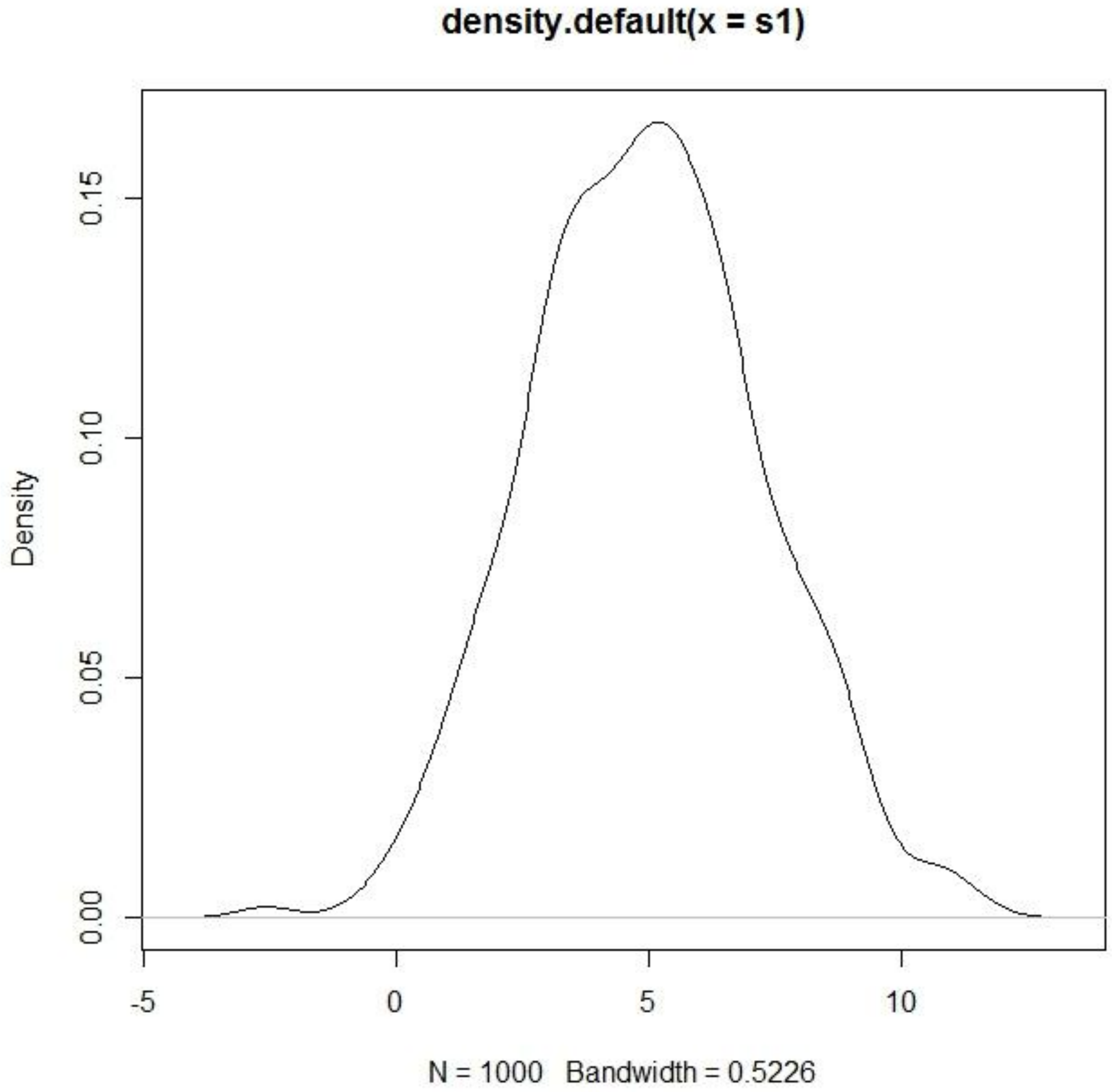
Implementation using R(using Box-Muller Method):

```
boxmuller<-function(n)
{
        RN<-NULL;
        for(i in 1:n)
        {

                u1<-runif(1,min=0,max=1);
                u2<-runif(1,min=0,max=1);
                r<--2*log(u1);
                v<-2*(3.14)*u2;
                z1<-(sqrt(r))*cos(v);
                z2<-(sqrt(r))*sin(v);
                RN<-c(RN,z1,z2);
        }
        return(RN);
}
s<-boxmuller(500);
s1<-5+s*sqrt(5);
print(s1);
```

Using the output the following histogram was generated:



The density plot of the numbers was like this:

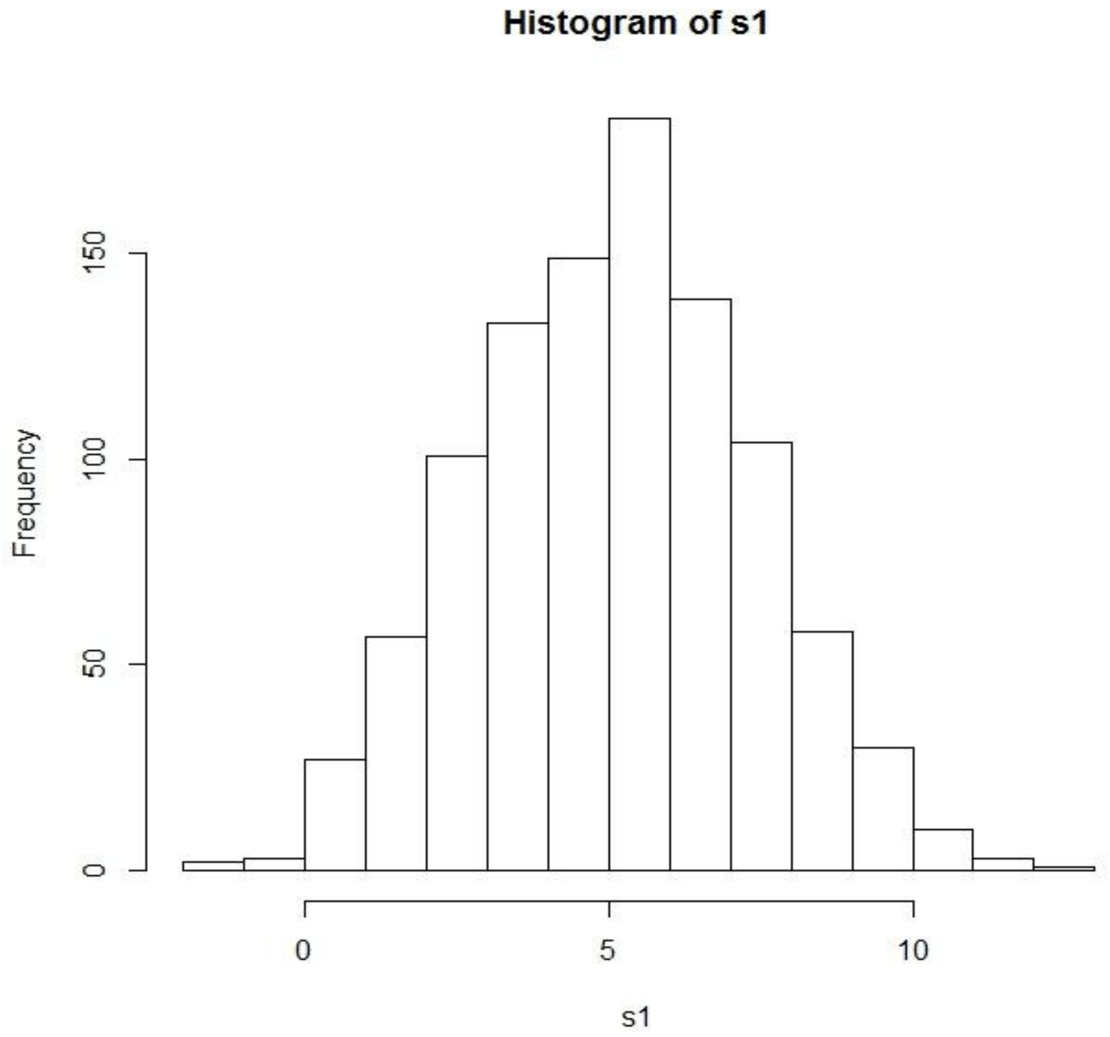


Implementation using R(using marsaglia variant):

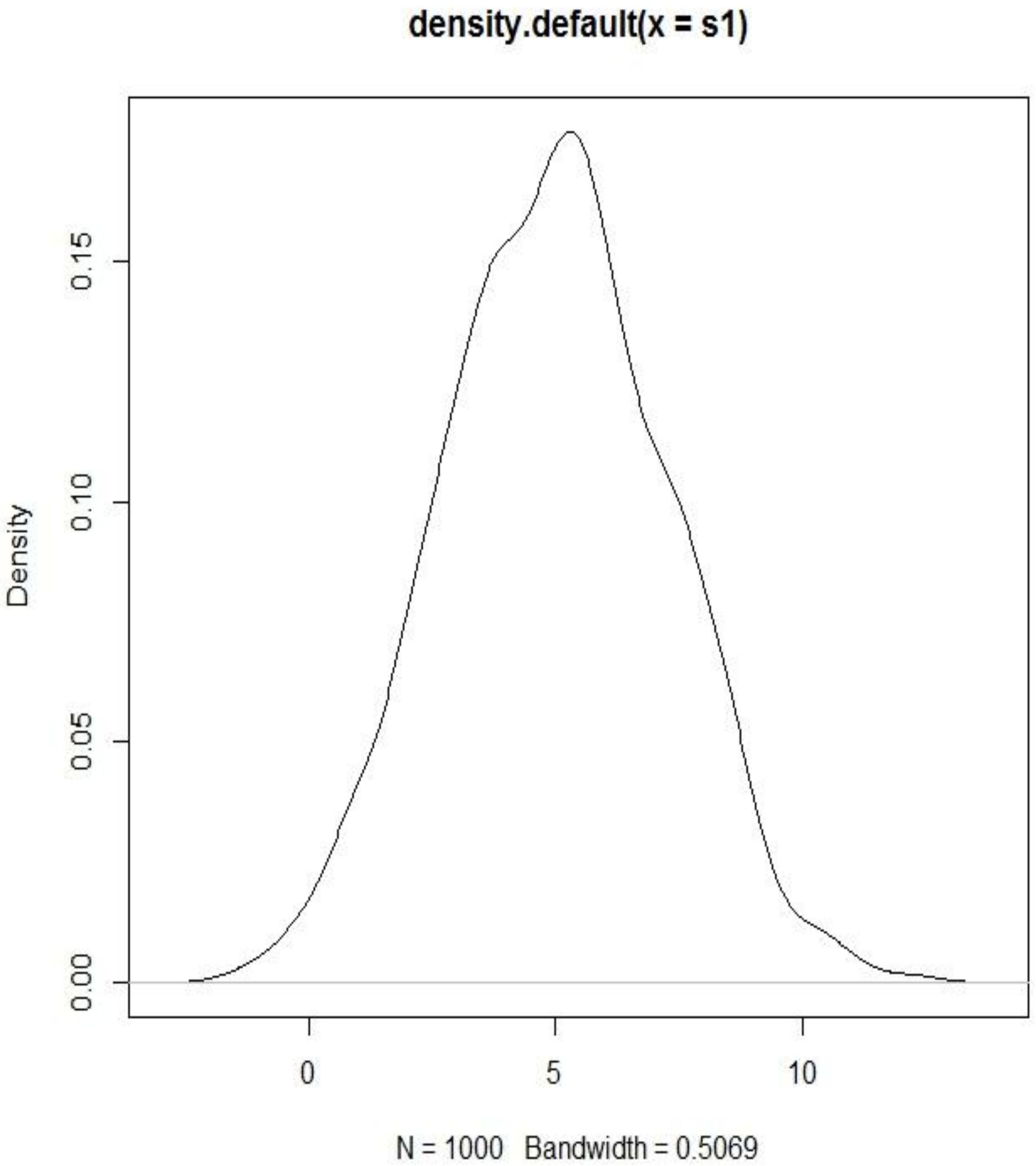
```
#this is the r implementation of the box-muller method with marsaglia-bray modifications
boxmuller<-function(n)
{
        RN<-NULL;
        for(i in 1:n)
        {
                u1<-0;
                u2<-0;
                k<-2;
                while(k>1)
                {
                        u1<-runif(1,min=0,max=1);
                        u1<-(2*u1)-1;
                        u2<-runif(1,min=0,max=1);
```

```
            u2<-(2*u2)-1;
            k<-(u1*u1)+(u2*u2);
        }
        #print(paste(k));
        r<-(-2)*log(k);
        cv<-u1/sqrt(k);
        sv<-u2/sqrt(k);
        z1<-sqrt(r)*cv;
        z2<-sqrt(r)*sv;
        RN<-c(RN,z1,z2);
    }
    return(RN);
}
s<-boxmuller(500);
s1<-5+s*sqrt(5);
print(s1);
```

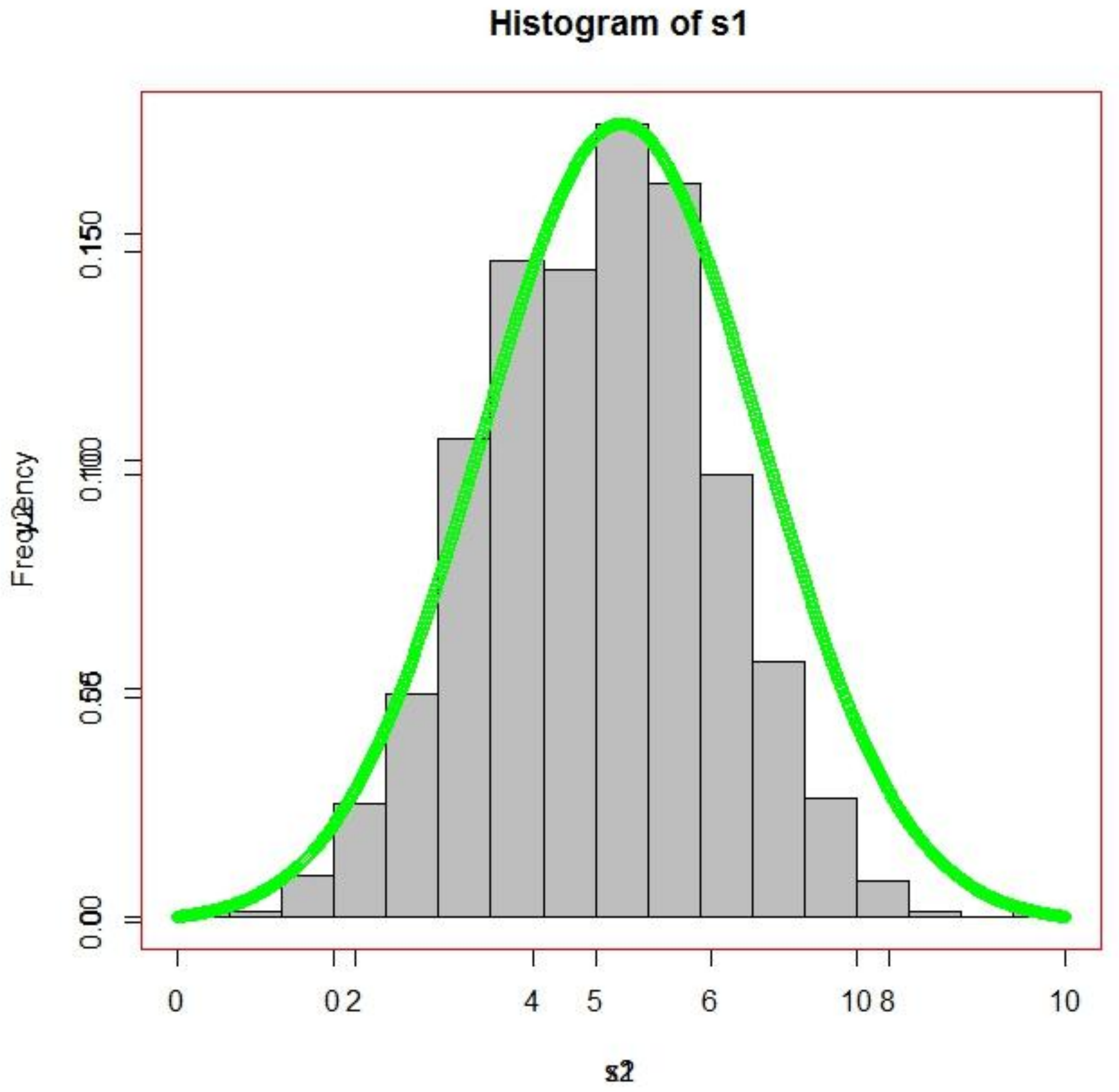
<u>The histogram generated by using the output:</u>



The density plot of the numbers generated:



Comparing with the standard normal curve generated theoretically:

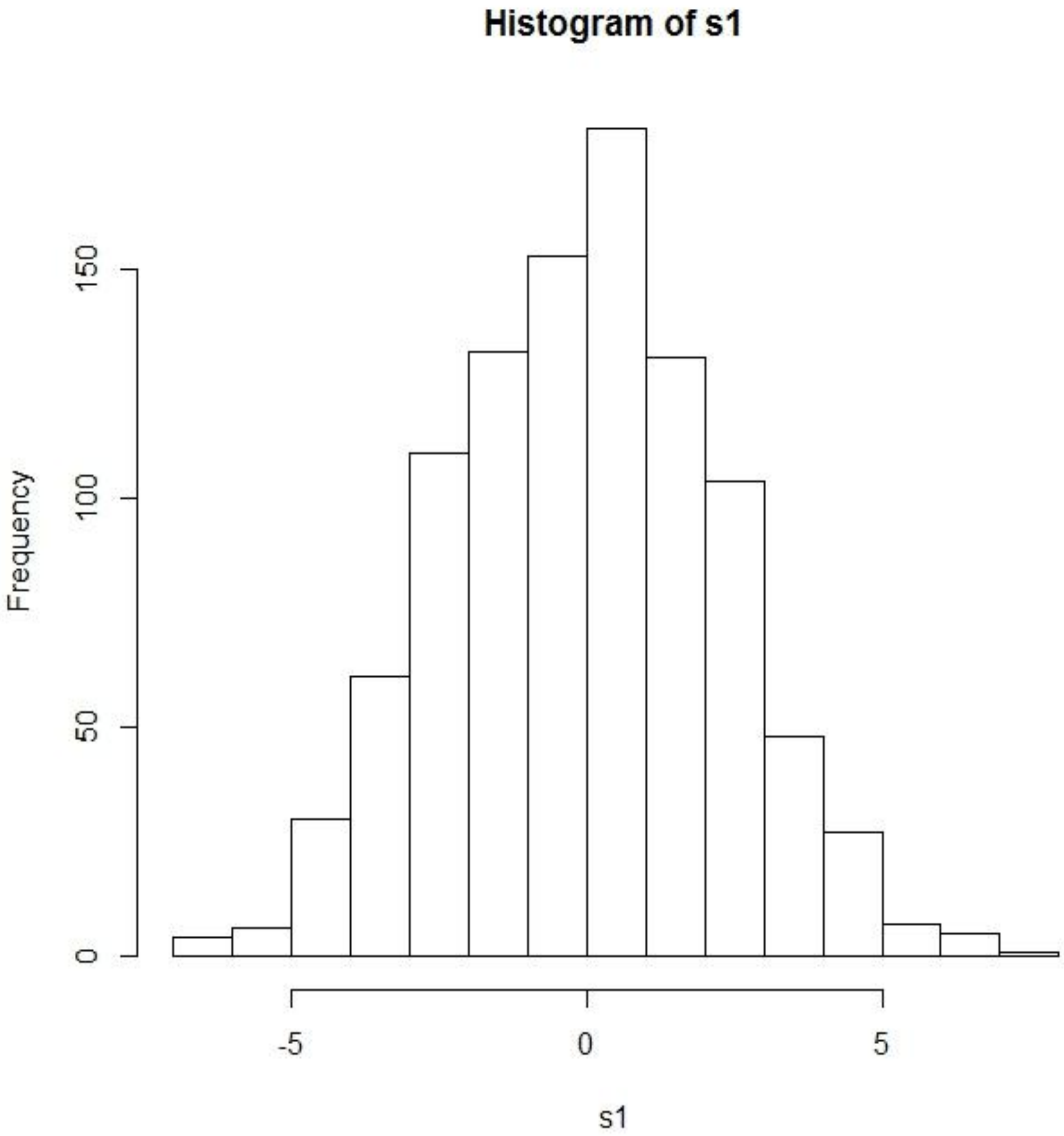


For N(µ = 0, б = 5):

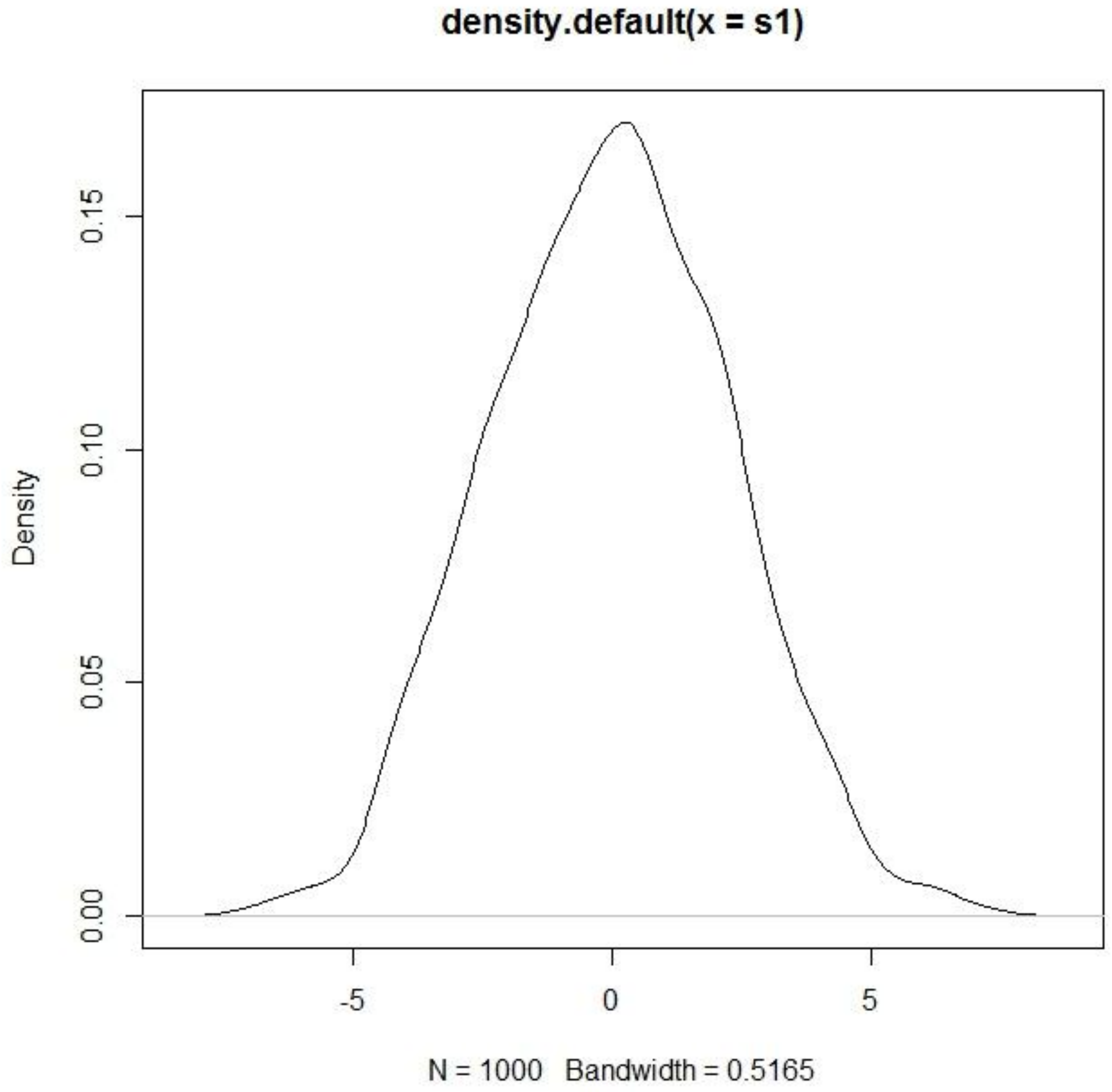
Implementation using R(using Box-Muller method):

```
boxmuller<-function(n)
{
        RN<-NULL;
        for(i in 1:n)
        {

                u1<-runif(1,min=0,max=1);
                u2<-runif(1,min=0,max=1);
                r<--2*log(u1);
                v<-2*(3.14)*u2;
                z1<-(sqrt(r))*cos(v);
                z2<-(sqrt(r))*sin(v);
                RN<-c(RN,z1,z2);
        }
        return(RN);
}
s<-boxmuller(500);
s1<-0+s*sqrt(5);
print(s1);
hist(s1, freq=TRUE);
```

<u>Using the output the following histogram was generated:</u>



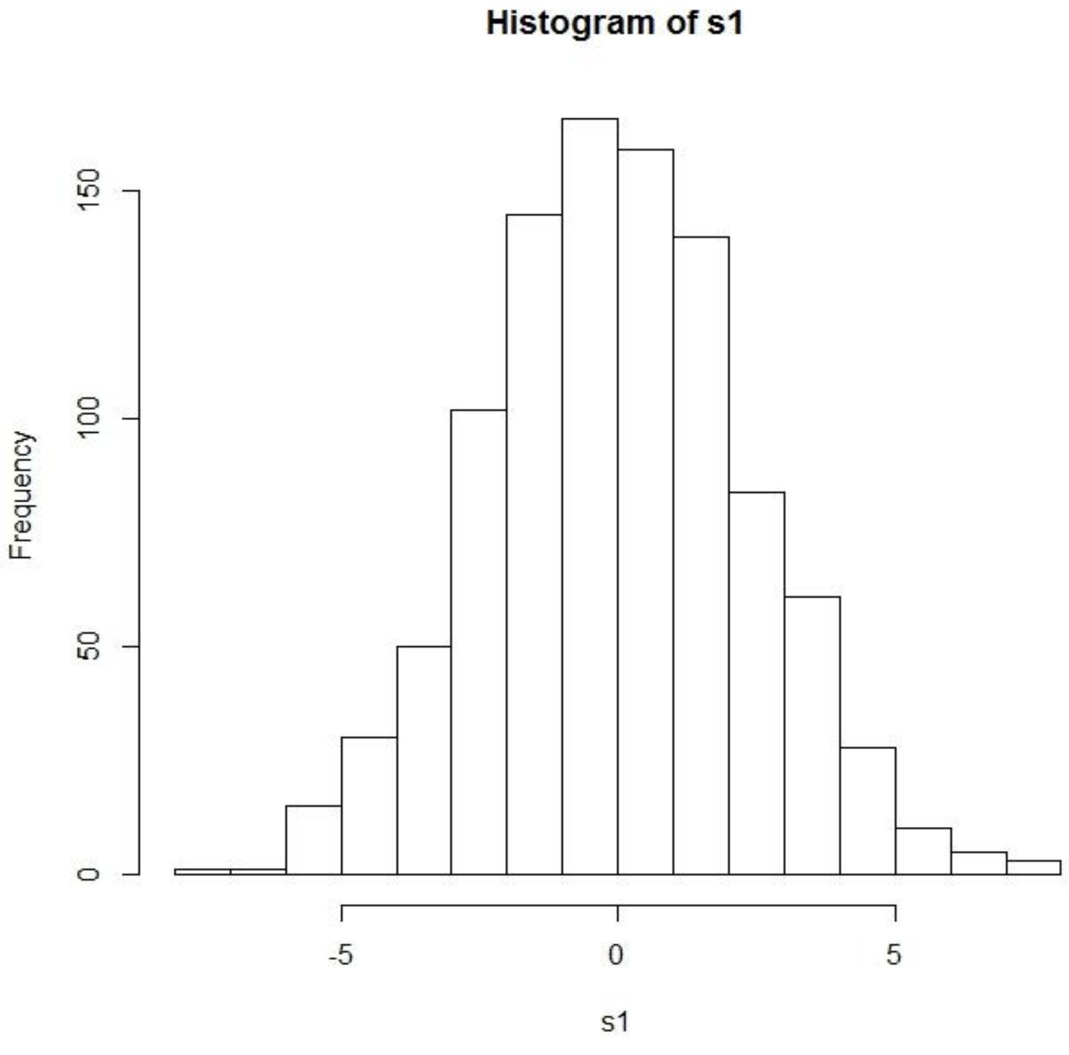
The Density plot the obtained numbers was like this:



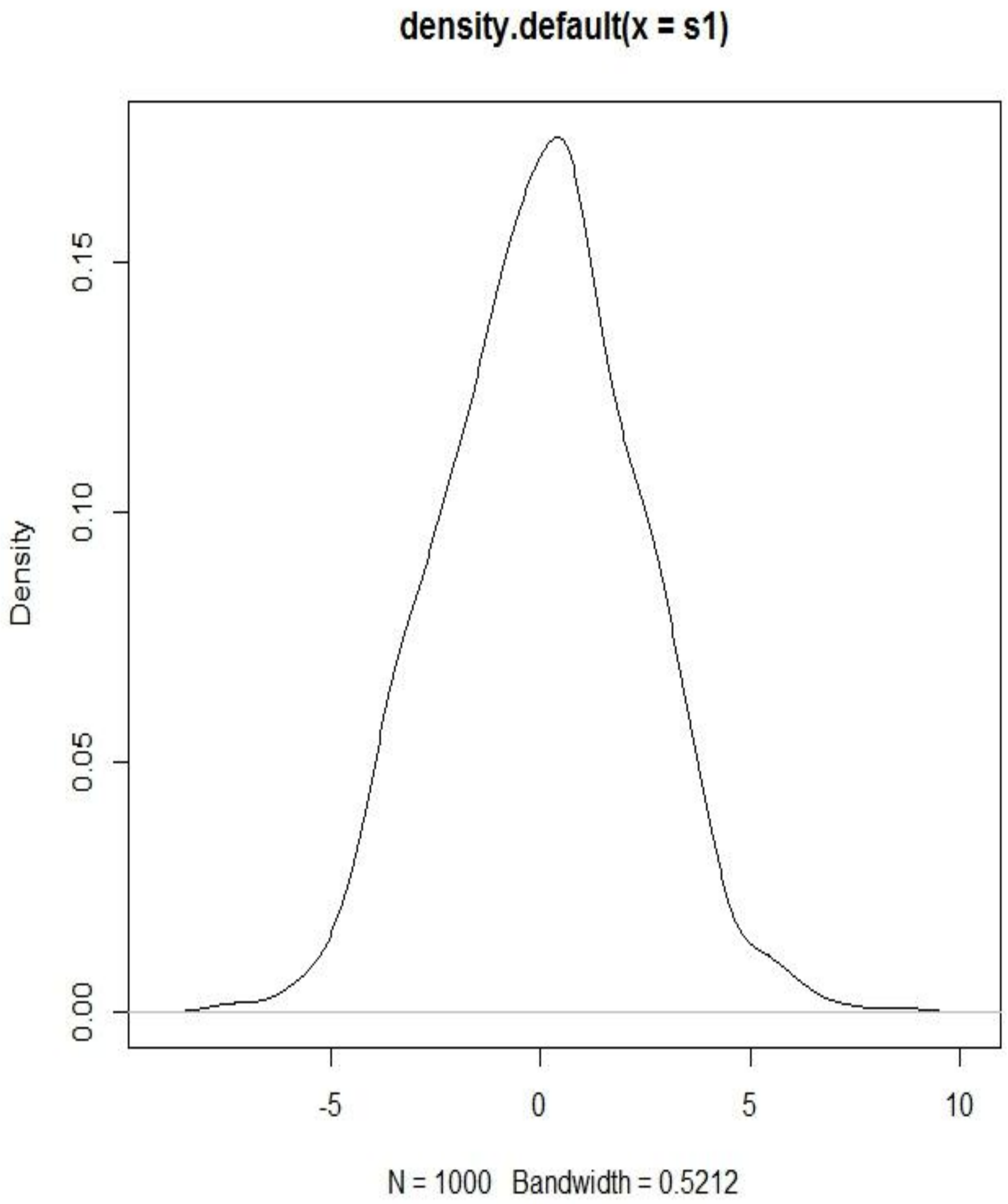
Implementation using R(using the marsaglia variant):

```
#this is the r implementation of the box-muller method with marsaglia-bray modifications
boxmuller<-function(n)
{
        RN<-NULL;
        for(i in 1:n)
        {
                u1<-0;
                u2<-0;
                k<-2;
                while(k>1)
                {
                        u1<-runif(1,min=0,max=1);
                        u1<-(2*u1)-1;
                        u2<-runif(1,min=0,max=1);
                        u2<-(2*u2)-1;
                        k<-(u1*u1)+(u2*u2);
                }
                #print(paste(k));
                r<-(-2)*log(k);
                cv<-u1/sqrt(k);
                sv<-u2/sqrt(k);
                z1<-sqrt(r)*cv;
                z2<-sqrt(r)*sv;
                RN<-c(RN,z1,z2);
        }
        return(RN);
}
s<-boxmuller(500);
s1<-0+s*sqrt(5);
print(s1);
```

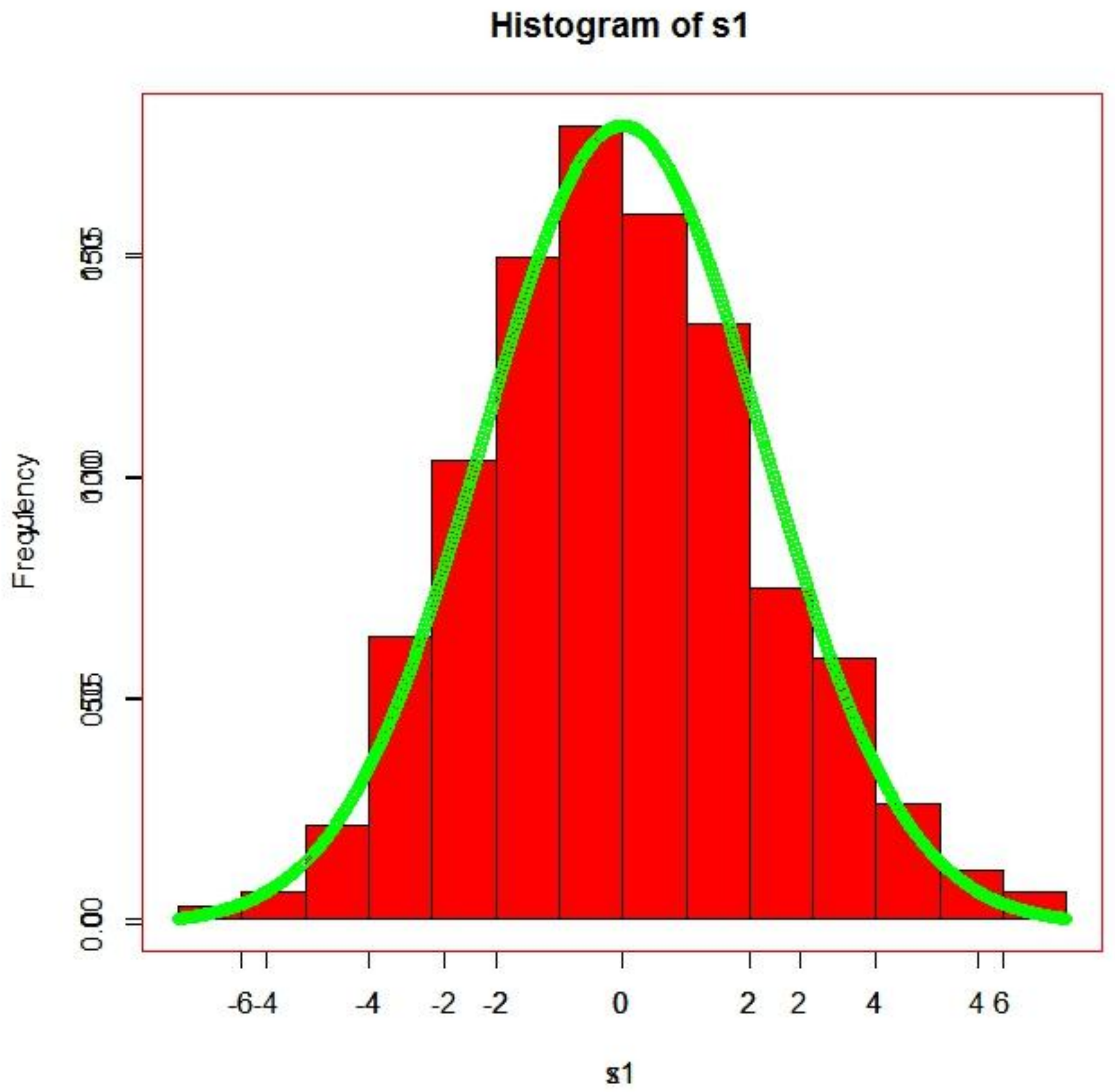
Using the output the following histogram was generated:



The density plot of the numbers was like this:



Comparing with the standard normal curve generated theoretically:



Conclusion:

The density plot on comparison with the standard normal distribution shows that the numbers generated follow normal distribution.

Part III:

Time taken by Marsaglia-Bray method is in this case: **0.83s** (for 500 numbers)

Time taken by Box-Muller method is in this case: **0.97s** (for 500 numbers)

Therefore the Marsaglia-Bray Method is more economic and faster.
(The time includes printing time for the values)

Tablulating some values:

Sample size	Box-Muller	Marsaglia Variant
1000	1.64s	1.63s
10000	19.83s	17.68s
20000	45.06s	39.24s
50000	137.86s	135.4s
100000	197.98s	183.7s
200000	501.92s	445.8s
500000	…………….	System failure!

Part IV:

This part calculates for the Marsaglia-Bray method the proportion of values rejected and accepted.

The R code given below calculate the acceptance ratio for us:

```
#this calculates the accepatance-rejection ratio
boxmuller<-function(n)
{
        RN<-NULL;
        val<-0;
        acc<-0;
        for(i in 1:n)
        {
                u1<-0;
                u2<-0;
                k<-2;
                while(k>1)
                {
                        u1<-runif(1,min=0,max=1);
                        u1<-(2*u1)-1;
                        u2<-runif(1,min=0,max=1);
                        u2<-(2*u2)-1;
                        k<-(u1*u1)+(u2*u2);
                        val<-val+1;
                }
                acc=acc+1;
                #print(paste(k));
                r<-(-2)*log(k);
                cv<-u1/sqrt(k);
                sv<-u2/sqrt(k);
                z1<-sqrt(r)*cv;
                z2<-sqrt(r)*sv;
                RN<-c(RN,z1,z2);
        }
        print(paste(acc));
        print(paste(val));
        print(paste(acc/val));
        return(RN);
}
s<-boxmuller(500);
#s1<-5+s*sqrt(5);
#print(s1);
```

<u>Result:</u>

Accepatance-Ratio: 0.7861635

Rejection ratio: 0.2138365

Thus, the rejection ratio is close to (1-π/4)=0.2146018.

Thus tablulating the results:-

Number of samples	Rejection ratio
100	0.2307692
500	0.2138365
10000	0.2142298