MA 226
Monte Carlo Simulation

Assignment - 2
(18-01-2013)

AMIT MITTAL

(Roll No.: 11012304)
Department of Mathematics
Indian Institute of Technology Guwahati

Assignment

All the following problems are for the following general linear congruence generator :

$$x_{i+1} = (ax_i + b) \, mod \, m$$

$$u_{i+1} = (x_{i+1})/m$$

1. Generate the sequence of numbers x_i for $a=6, b=0, m=11,$ and x_0 ranging from 0 to 10. Also, generate the sequence of numbers x_i for $a=3, b=0, m=11,$ and x_0 ranging from 0 to 10. Observe the sequence of numbers generated and observe the repetition of values. Tabulate these for each group of values. How many distinct values are appearing before repetitions? Which, in your view, are the best choices and why?
(Use only C or C++ code)

2. Generate a sequence u_i with $m=244944, a=1597, 51749$ (take x_0 as per your choice). Try to group the values in the ranges $0-0.05, 0.05-0.10, 0.10-0.15, \ldots$ and see their frequencies (i.e. the number of values falling in a group). For at least 5 different values of the number of values generated, tabulate the frequencies in each case, draw bar diagrams of these data and put in your observations.
(Use both R and C / C ++ code)

3. Generate a sequence u_i with $a=1229, b=1, m=2048$. Plot in a two-dimensional graph the points (u_{i-1}, u_i) i.e., the points $(u_1, u_2), (u_2, u_3), (u_3, u_4), \ldots$ What are your observations?

Question-I

All the following problems are for the following general linear congruence generator :

$x_{i+1} = (a*x_i + b) \bmod m$

$u_{i+1} = x_{i+1} / m$

Generate the sequence of numbers x_i for a = 6, b = 0, m = 11 and x_0 ranging from 0 to 10. Also, generate the sequence of numbers x_i for a = 3, b = 0, m = 11, and x_0 ranging from 0 to 10. Observe the sequence of numbers generated and observe the repetition of values. Tabulate these for each group of values. How many distinct values are appearing before repetitions? Which, in your view, are the best choices and why?

C++ CODE :

```
#include <iostream>
#include <cstdio>

using namespace std;

double func_rand(int x, int M){
	return (double)x/M;
}

int func_val(int prev, int A, int B, int M){
	int next;
	next = ((A*prev) + B)%M;

	return next;
}

int main(){
	int init, val;

	for(init = 0 ; init < 11 ; ++init){
		val=init;
		printf("x%d=%d\t",init, init);
		do{
			printf("%d\t", val);
			val = func_val(val, 6, 0, 11);
		}while(val!=init);
		printf("%d\n", init);
	}
	printf("\n");

	for(init = 0 ; init < 11 ; ++init){
		val=init;

		printf("x%d=%d\t",init, init);
		do{
			printf("%d\t", val);
			val = func_val(val, 3, 0, 11);
		}while(val!=init);

		printf("%d\n", init);
	}
	return 0;
}
```

OUTPUT

a=6 , b=0 , m=11:

X_i → / X_0 ↓	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
X_0=0	0									
X_0=1	1	6	3	7	9	10	5	8	4	2
X_0=2	2	1	6	3	7	9	10	5	8	4
X_0=3	3	7	9	10	5	8	4	2	1	6
X_0=4	4	2	1	6	3	7	9	10	5	8
X_0=5	5	8	4	2	1	6	3	7	9	10
X_0=6	6	3	7	9	10	5	8	4	2	1
X_0=7	7	9	10	5	8	4	2	1	6	3
X_0=8	8	4	2	1	6	3	7	9	10	5
X_0=9	9	10	5	8	4	2	1	6	3	7
X_0 10	10	5	8	4	2	1	6	3	7	9

a=3 , b=0 , m=11:

X_i → / X_0 ↓	X_1	X_2	X_3	X_4	X_5
X_0=0	0				
X_0=1	1	3	9	5	4
X_0=2	2	6	7	10	8
X_0=3	3	9	5	4	1
X_0=4	4	1	3	9	5
X_0=5	5	4	1	3	9
X_0=6	6	7	10	8	2
X_0=7	7	10	8	2	6
X_0=8	8	2	6	7	10
X_0=9	9	5	4	1	3
X_0 10	10	8	2	6	7

OBSERVATIONS

For $a=6, b=0, m=11,$ and the value of x_0 = 0, the period is 1 i.e. x_i always remain 0. And for each x_0 varying from 1 to 10, the period is 10 i.e. there are 10 distinct values in the sequence after which the sequence starts repeating.

And for $a=3, b=0, m=11,$ and the value of x_0 = 0, the period is 1 i.e. x_i always remain 0. And for each x_0 varying from 1 to 10, the period is 5 i.e. there are 5 distinct values in the sequence after which the sequence starts repeating.

RESULT

For the case when $a=6, b=0, m=11,$ more random values are generated and also they are more uniformly distributed, therefore it is the better choice.

QUESTION-II

Generate a sequence u_i with m = 244944, a = 1597, 51749 (take x_0 as per your choice). Try to group the values in the ranges 0 - 0.05, 0.05 - 0.10, 0.10 - 0.15, ... and see their frequencies (i.e. the number of values falling in a group). For at least 5 different values
of the number of values generated, tabulate the frequencies in each case, draw bar diagrams of these data and put in your observations. (Use both R and C / C ++ code)

C++ CODE :

```
#include <iostream>
#include <cstdio>

#define LL long long

using namespace std;

double func_rand(LL int val, LL int m){
	return (double)val/m;
}

LL int func_val(LL int val, LL int a, LL int b, LL int m){
	LL next_val;
	next_val = ((a * val) + b) % m;

	return next_val;
}

void function(LL int initial, LL int a, LL int b, LL int m, int end){
	LL int val, count = 0, array[20] = {0};
	int index, quot;
	double random;

	val = initial;
	random = func_rand(val, m);

	while(count<end){
		val = func_val(val, a, b, m);
		random = func_rand(val, m);

		quot = random/0.05;
		array[quot]++;
		++count;
	}

	for(index = 0 ; index<20 ; ++index){
		printf("%lf - %lf\t%lld\n", 0.05*index, 0.05*(index+1), array[index]);
	}
}


int main(){
	LL int index, a[5] = {100, 1000, 2000, 4000, 5000};
```

```
	LL int init_values[5] = {5, 73, 128, 23451, 2341};
	LL int bound = 1000;

	for(index = 0; index<5; ++index){
		printf("x0=%lld\n", init_values[index]);
		function(init_values[index], 1597, 0, 244944, bound);
		cout<<endl;
	}

	cout<<"x0 = "<<init_values[4]<<endl;
	for(index = 0; index < 5; ++index){
		printf("Bound = %lld\n", a[index]);
		function(init_values[4], 1597, 0, 244944, a[index]);
		printf("\n");
	}

	printf("\n\n\n");

	for(index = 0; index<5; ++index){
		printf("x0=%lld\n", init_values[index]);
		function(init_values[index], 51749, 0, 244944, bound);
		cout<<endl;
	}

	cout<<"x0 = "<<init_values[4]<<endl;
	for(index = 0; index < 5; ++index){
		printf("Bound = %lld\n", a[index]);
		function(init_values[4], 51749, 0, 244944, a[index]);
		printf("\n");
	}

	return 0;
}
```

R CODE :

```
func_rand <- function(val, m){
	return(val/m);
}

func_val <- function(val, a, b, m){
	return(((a*val)+b)%%m);
}

func <- function(initial, a, b, m, end){
	count<-1;
	arr<-array(20);
	tab<-array();

	for(i in 1:20)
	{
		arr[i]=0
	}


	val <- initial;
	random <- func_rand(val, m);

	while(count<=end){
		val <- func_val(val, a, b, m);
		random <- func_rand(val, m);

		tab[count] = random;

		quot <- round(random/0.05);

		arr[quot] <- arr[quot] + 1;
		count <- count+1;
	}

	max<-0;
	for(i in 1:20){
		cat(sprintf("arr[%d]\t%d\n", i, arr[i]));

		if(arr[i]>max){
			max=arr[i];
		}
```

```
	}

	bins<-seq(0, 1, by=0.05);
	hist(tab, bins,freq = TRUE, include.lowest = TRUE, right = TRUE, density = NULL,
		angle = 45, col = '#1E90FF', border = NULL,
		main = paste("Histogram of frequencies for\nxo=", initial, " and total values
		generated=",end), xlim=range(0,1), ylim=range(0,(max+15)), xlab = arr ,
		axes = TRUE, plot = TRUE, labels = FALSE);

	dev.copy(jpeg,paste('a=',a,'x0=',initial,'bound=',end,sep="",".jpg"));
	dev.off();
}

main <- function(){
	a<-c(100, 1000, 2000, 4000, 5000);
	init_values<-c(5, 73, 128, 23451, 2341);
	bound<-1000;

	for(index in 1:5){
		func(init_values[index], 1597, 0, 244944, bound);
	}


	for(index in 1:5){
		func(init_values[4], 1597, 0, 244944, a[index]);
	}

	cat("\n\n\n");

	for(index in 1:5){
		func(init_values[index], 51749, 0, 244944, bound);
	}

	for(index in 1:5){
		func(init_values[4], 51749, 0, 244944, a[index]);
	}
}
```

OUTPUT

Frequency table for $m=244944, a=1597, b=0,$:

X_0 →	$X_0 = 5$	$X_0=73$	$X_0=128$	$X_0=23451$	$X_0=2341$
Intervals ↓					
(0.00-0.05)	40	38	50	49	56
(0.05-0.10)	46	55	53	52	46
(0.10-0.15)	50	49	46	49	45
(0.15-0.20)	43	59	45	50	46
(0.20-0.25)	54	55	51	48	52
(0.25-0.30)	47	43	51	51	56
(0.30-0.35)	51	50	53	53	57
(0.35-0.40)	50	57	49	51	46
(0.40-0.45)	46	44	48	51	56
(0.45-0.50)	49	48	55	49	57
(0.50-0.55)	63	53	53	50	35
(0.55-0.60)	51	48	52	50	57
(0.60-0.65)	54	53	51	48	49
(0.65-0.70)	58	48	47	50	52
(0.70-0.75)	53	47	49	48	47
(0.75-0.80)	60	56	46	53	56
(0.80-0.85)	52	48	54	50	51
(0.85-0.90)	44	42	48	49	57
(0.90-0.95)	46	53	53	51	42
(0.95-1.00)	43	49	46	48	43

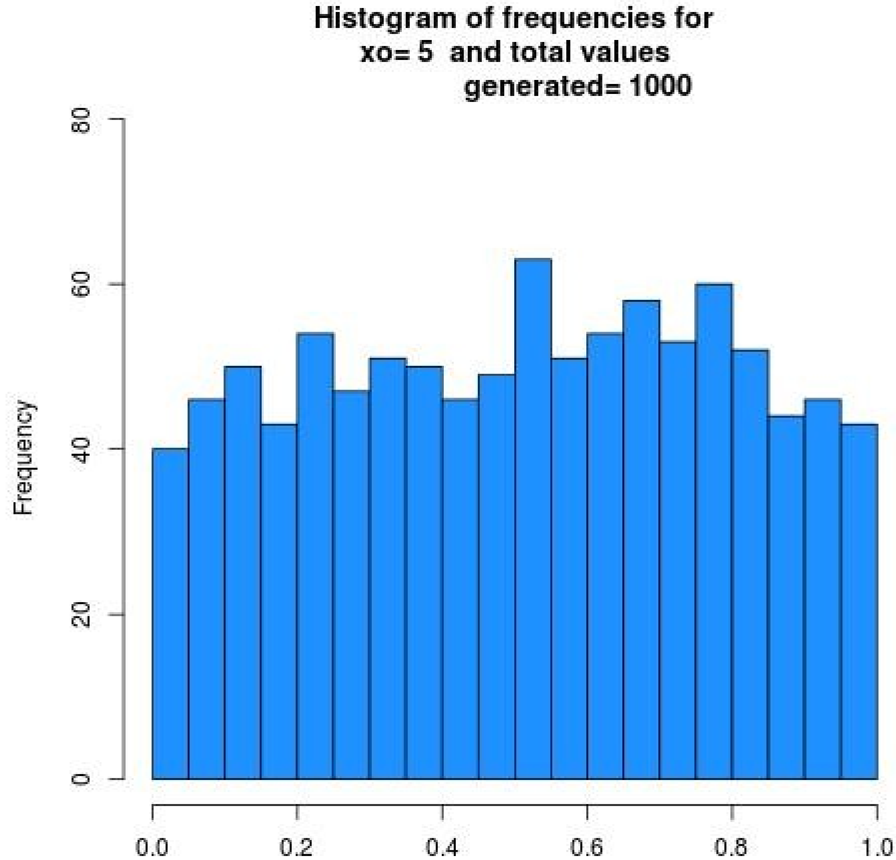
Frequency table for $m=244944, a=1597, b=0,$

The x_0 I have chosen to fix is $x_0=2341$
Here N denotes the number of elements the sequence starting with X_0

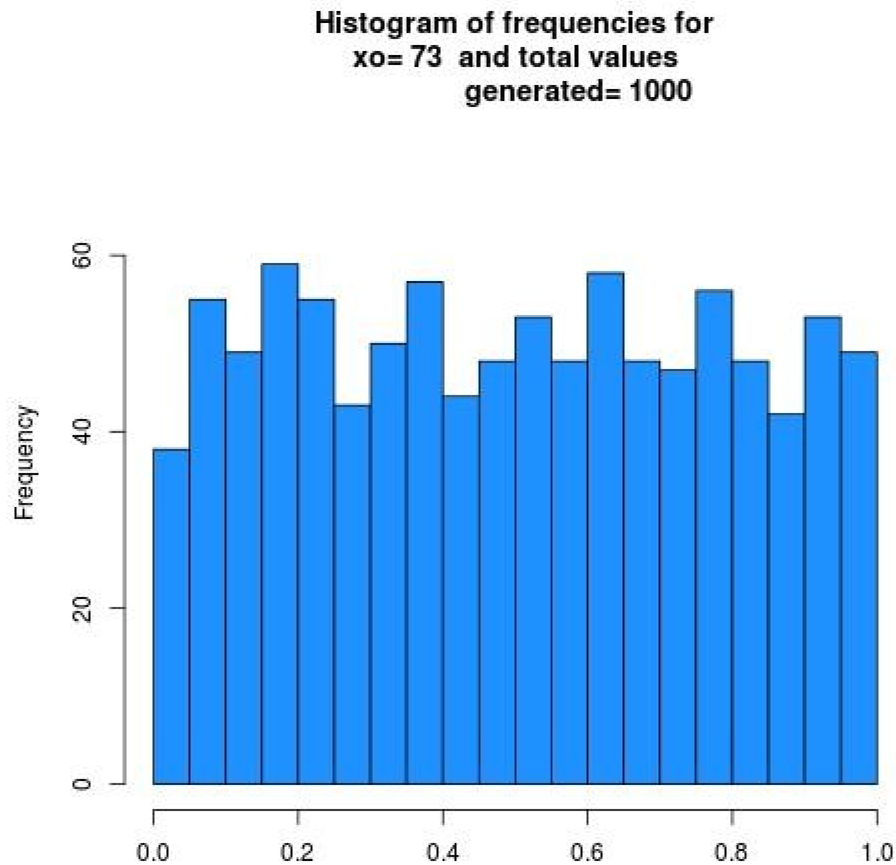
N →	**N=100**	**N=1000**	**N=2000**	**N=4000**	**N=5000**
Intervals ↓					
(0.00-0.05)	7	56	100	209	247
(0.05-0.10)	3	46	95	196	248
(0.10-0.15)	4	45	96	196	245
(0.15-0.20)	5	46	104	196	256
(0.20-0.25)	5	52	107	203	257
(0.25-0.30)	3	56	96	205	247
(0.30-0.35)	3	51	102	201	251
(0.35-0.40)	3	46	100	195	253
(0.40-0.45)	3	56	101	206	250
(0.45-0.50)	5	57	101	205	251
(0.50-0.55)	5	35	87	184	238
(0.55-0.60)	7	57	107	206	256
(0.60-0.65)	9	49	109	197	257
(0.65-0.70)	8	52	96	199	247
(0.70-0.75)	7	47	95	197	245
(0.75-0.80)	3	56	112	202	261
(0.80-0.85)	6	51	99	203	250
(0.85-0.90)	7	57	103	212	253
(0.90-0.95)	6	42	93	193	244
(0.95-1.00)	1	43	97	195	244

HISTOGRAMS FOR THE FIRST FREQUENCY TABLE:

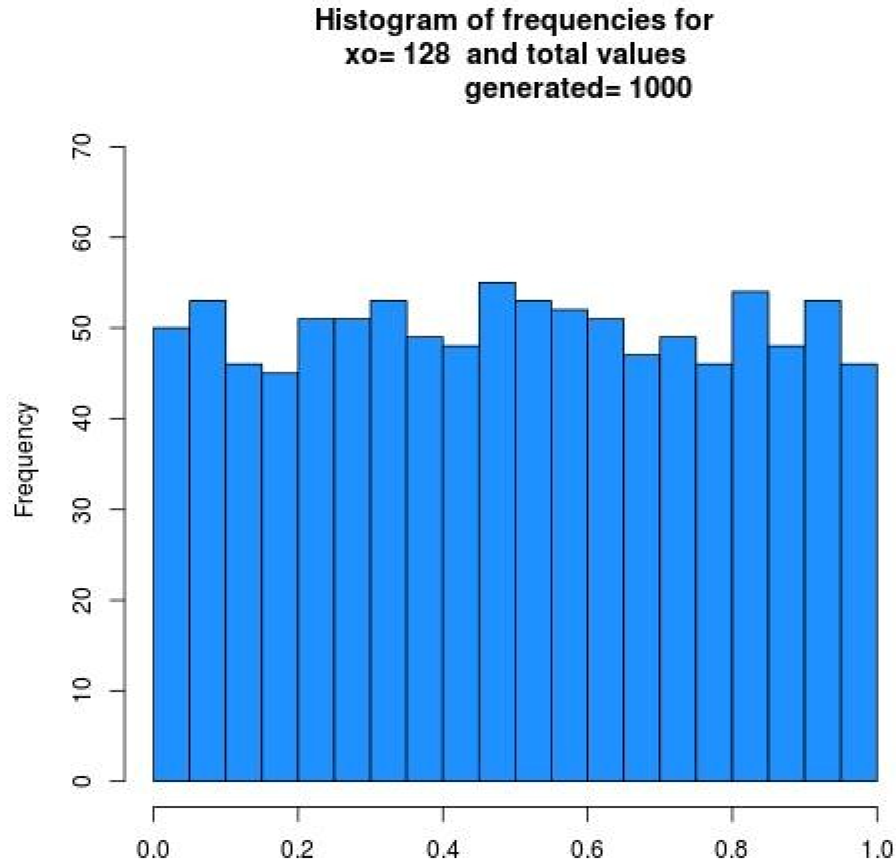
$X_0 = 5$
a=1597
b=0
m=244944
n=1000



$X_0 = 73$
a=1597
b=0
m=244944
n=1000

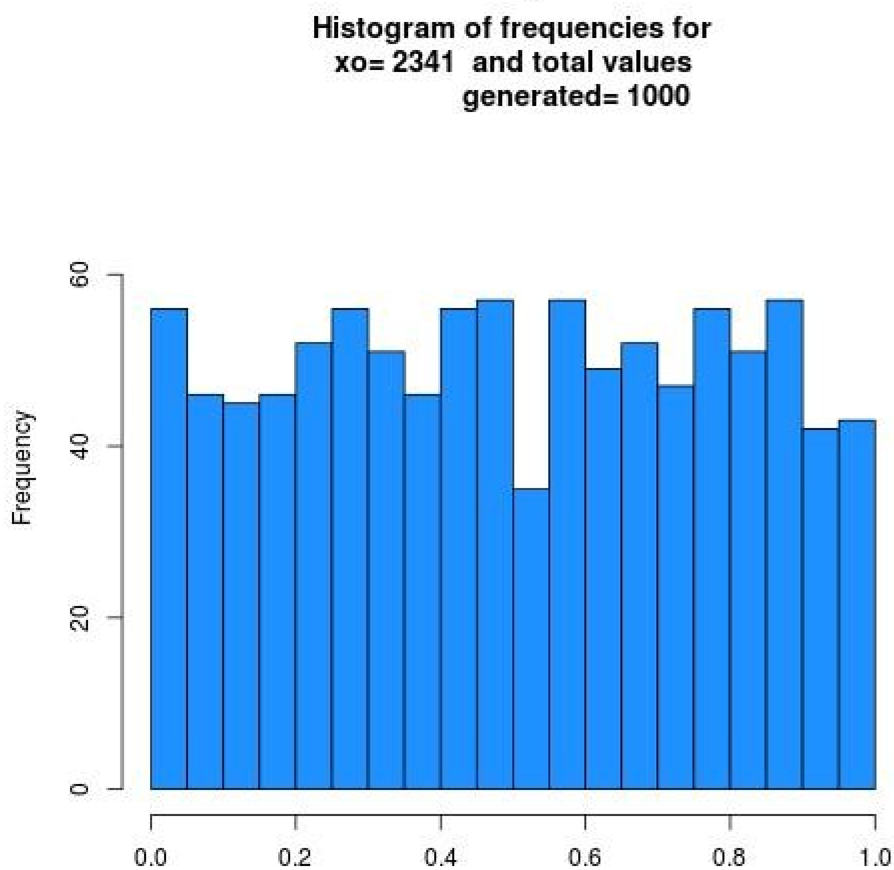


X_0 =128
a=1597
b=0
m=244944
n=1000



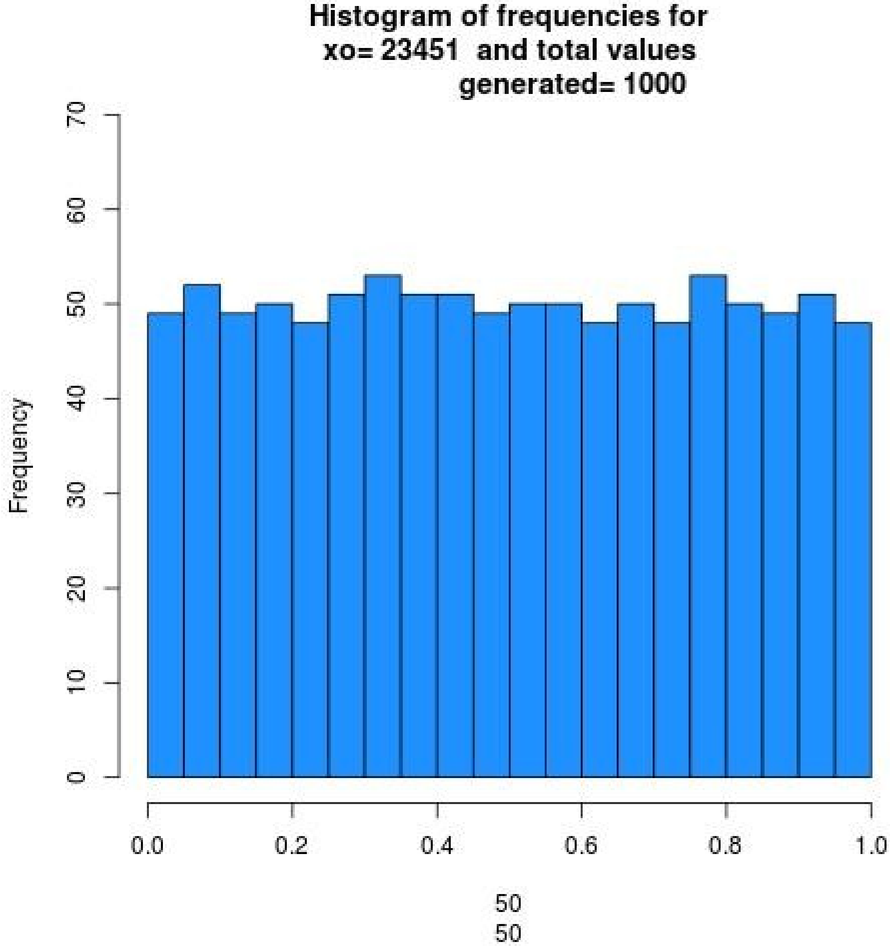
50

X_0 =2341
a=1597
b=0
m=244944
n=1000



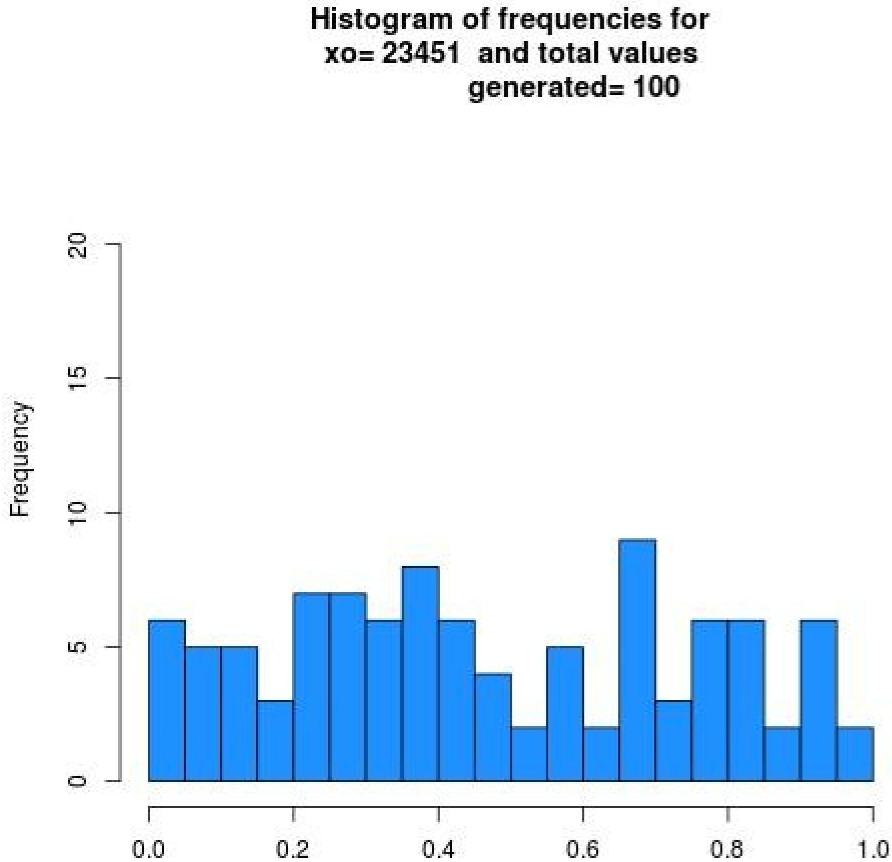
52
37

X_0 =23451
a=1597
b=0
m=244944
n=1000

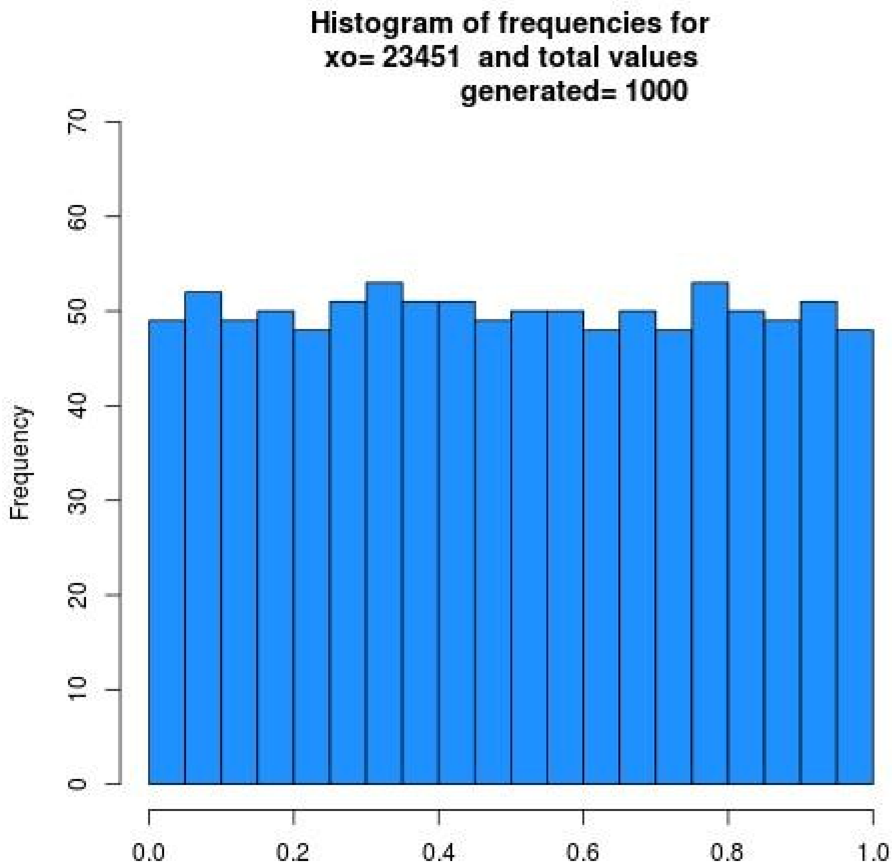


HISTOGRAMS FOR THE SECOND FREQUENCY TABLE:

X_0=23451
a=1597
b=0
m=244944
N=100

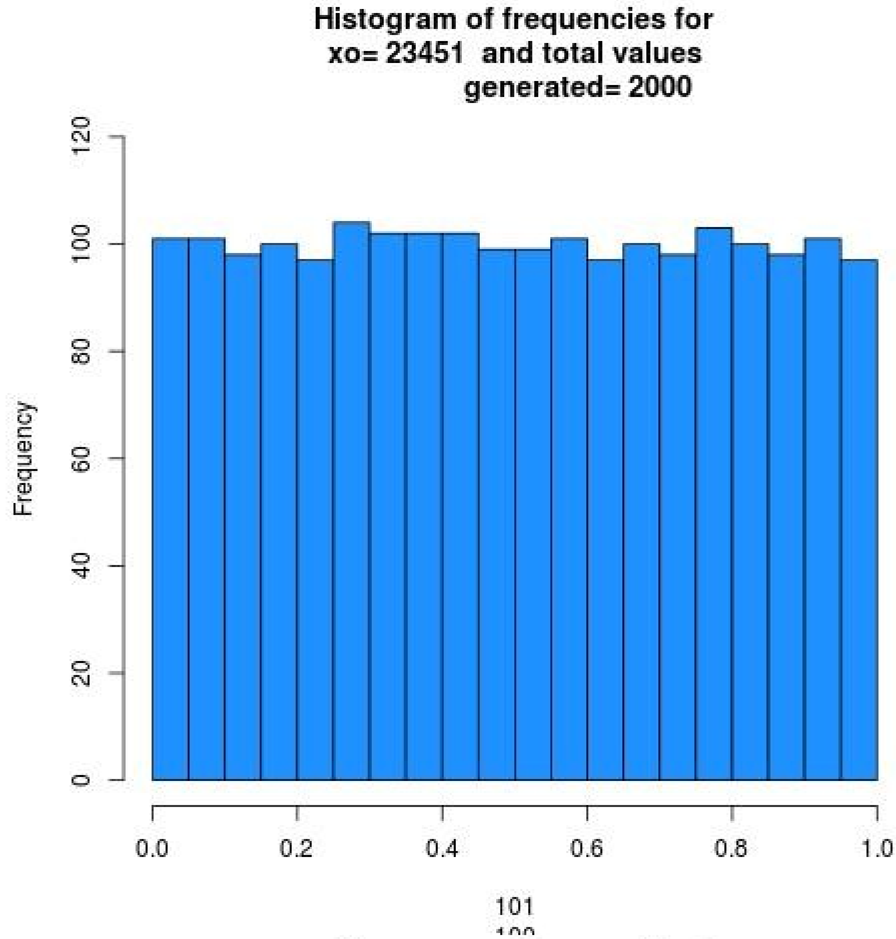


X_0=23451
a=1597
b=0
m=244944
N=1000

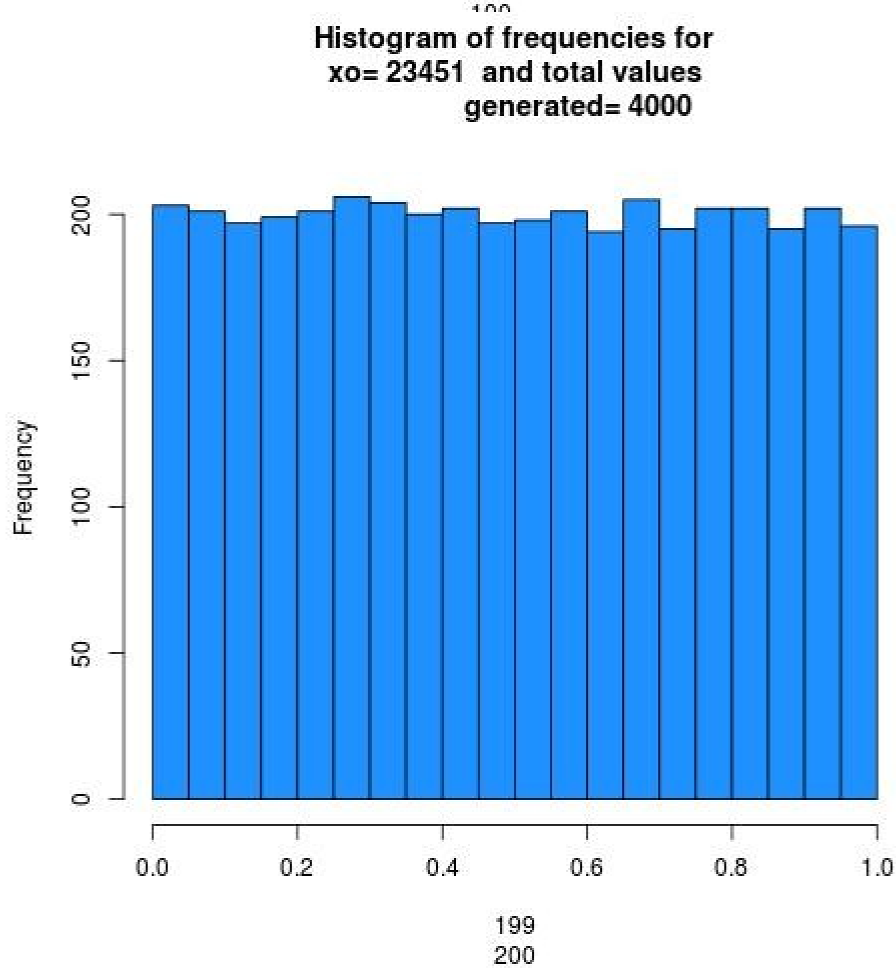


50

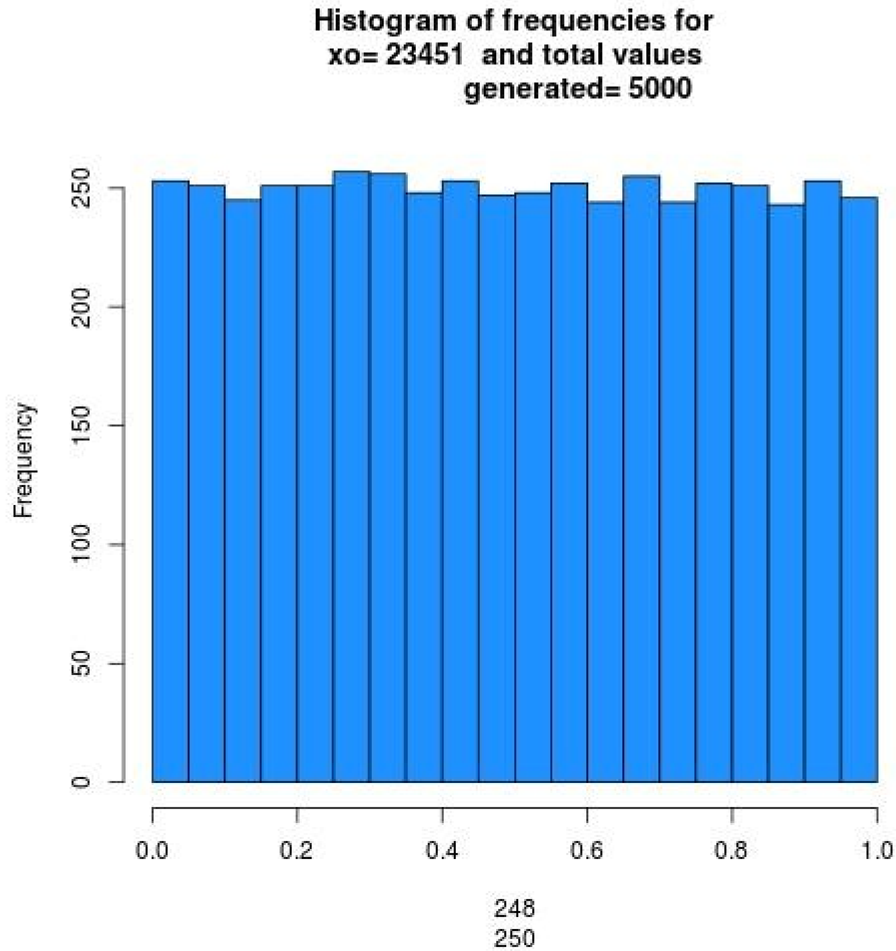
X_0=23451
a=1597
b=0
m=244944
N=2000



X_0=23451
a=1597
b=0
m=244944
N=4000



X_0=23451
a=1597
b=0
m=244944
N=5000



OUTPUT

Frequency table for $m=244944, a=51749, b=0,$:

X_0 →	$X_0 = 5$	$X_0=73$	$X_0=128$	$X_0=23451$	$X_0=2341$
Intervals ↓					
(0.00-0.05)	50	53	55	50	52
(0.05-0.10)	50	48	51	48	51
(0.10-0.15)	49	50	50	52	51
(0.15-0.20)	48	49	50	50	48
(0.20-0.25)	50	50	49	46	48
(0.25-0.30)	50	51	49	56	50
(0.30-0.35)	50	50	50	45	50
(0.35-0.40)	50	51	49	50	51
(0.40-0.45)	52	50	51	52	49
(0.45-0.50)	49	48	50	52	50
(0.50-0.55)	49	52	50	51	48
(0.55-0.60)	49	50	52	49	52
(0.60-0.65)	52	52	49	53	50
(0.65-0.70)	48	49	48	48	49
(0.70-0.75)	51	51	49	47	48
(0.75-0.80)	51	48	49	56	52
(0.80-0.85)	52	51	50	46	49
(0.85-0.90)	51	49	48	48	52
(0.90-0.95)	50	49	50	51	49
(0.95-1.00)	49	49	51	50	51

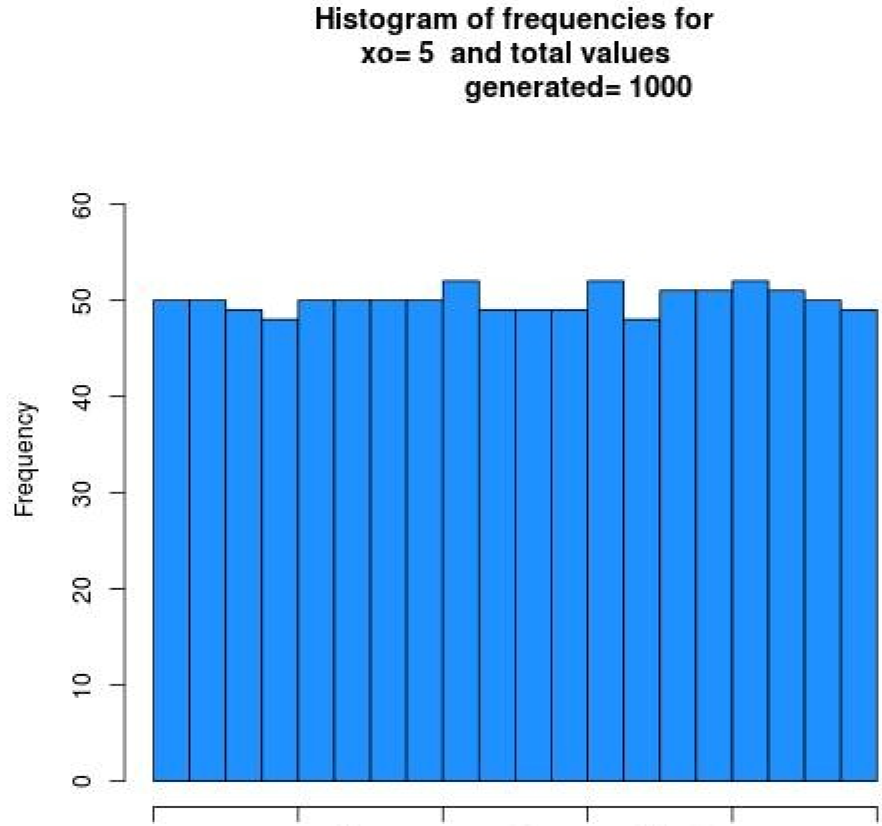
Frequency table for $m=244944, a=51749, b=0,$

The x_0 I have chosen to fix is $x_0=2341$
Here N denotes the number of elements the sequence starting with X_0

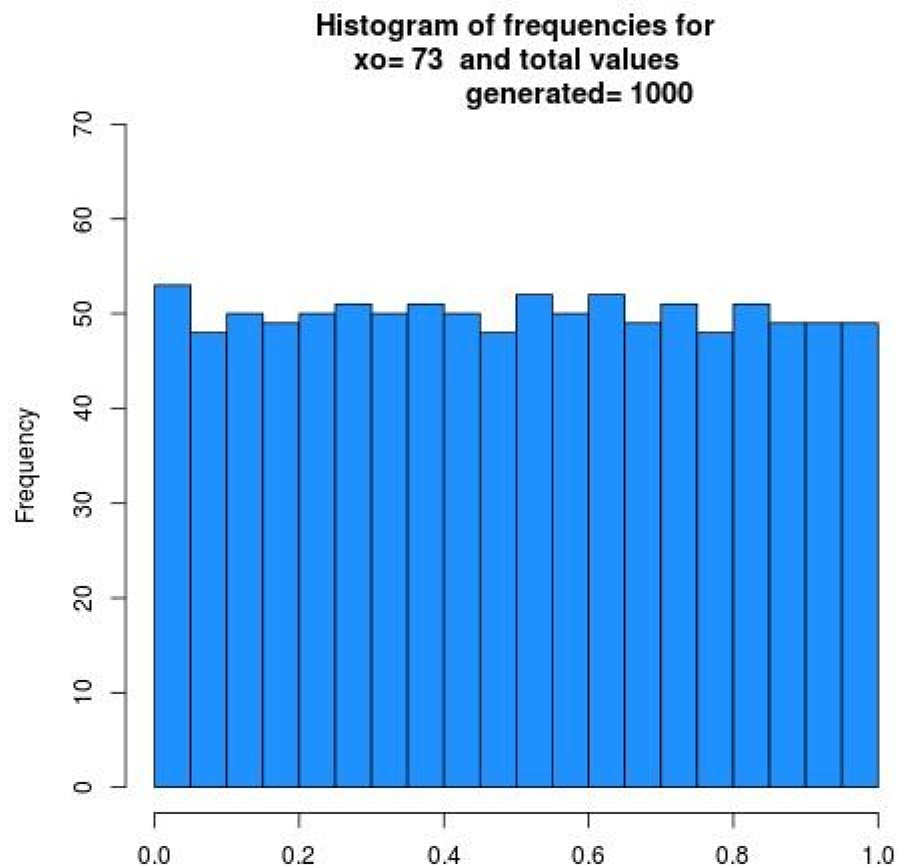
N →	**N=100**	**N=1000**	**N=2000**	**N=4000**	**N=5000**
Intervals ↓					
(0.00-0.05)	11	52	102	205	254
(0.05-0.10)	15	51	106	211	260
(0.10-0.15)	3	51	100	202	255
(0.15-0.20)	4	48	99	196	245
(0.20-0.25)	3	48	97	196	244
(0.25-0.30)	4	50	103	204	255
(0.30-0.35)	5	50	99	197	247
(0.35-0.40)	5	51	101	201	250
(0.40-0.45)	6	49	97	199	248
(0.45-0.50)	3	50	100	200	252
(0.50-0.55)	4	48	99	196	245
(0.55-0.60)	4	52	101	201	251
(0.60-0.65)	4	50	100	200	249
(0.65-0.70)	3	49	98	198	249
(0.70-0.75)	6	48	100	198	246
(0.75-0.80)	5	52	103	205	257
(0.80-0.85)	4	49	97	196	244
(0.85-0.90)	4	52	101	200	252
(0.90-0.95)	4	49	97	196	245
(0.95-1.00)	3	51	100	199	252

<u>**HISTOGRAM FOR THE FIRST FREQUENCY TABLE:**</u>

$X_0 = 5$
a=51749
b=0
m=244944
n=1000

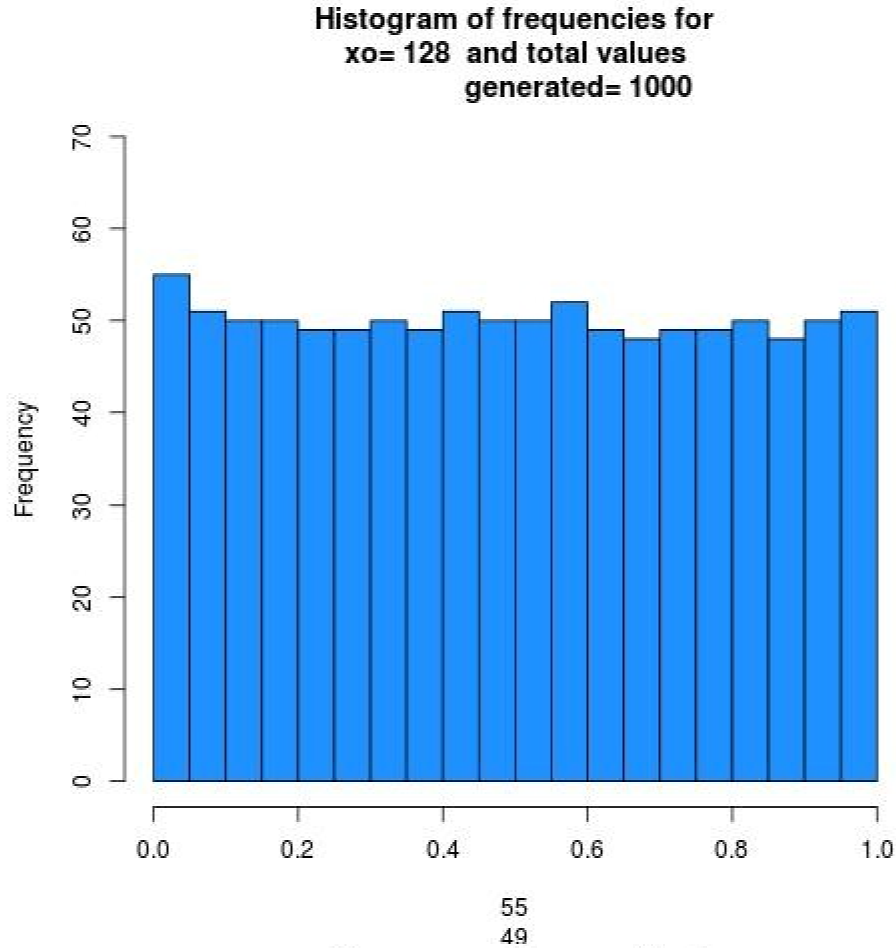


$X_0 = 73$
a=51749
b=0
m=244944
n=1000

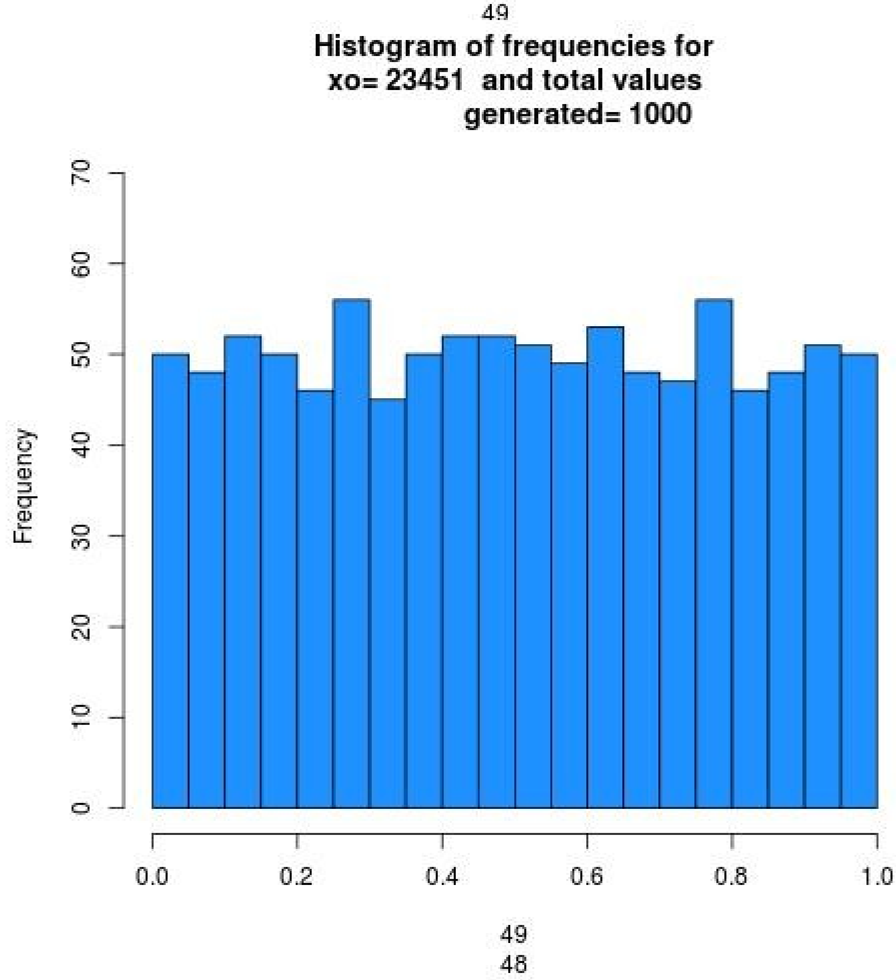


50

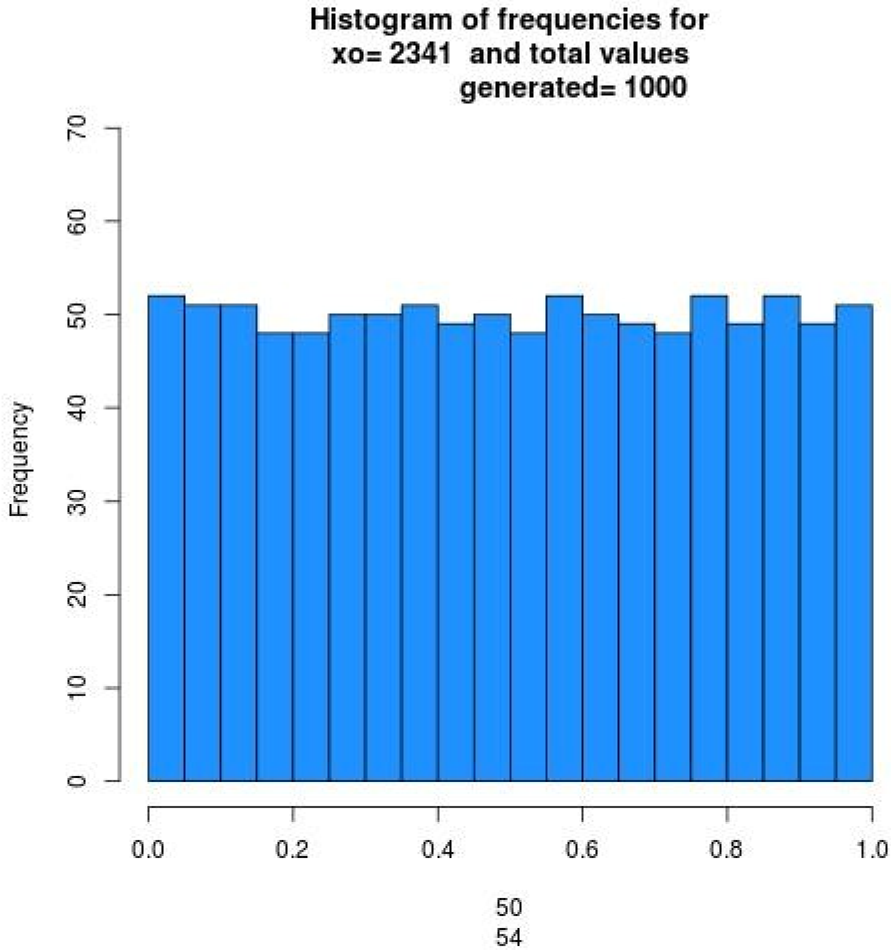
X_0 =128
a=51749
b=0
m=244944
n=1000



X_0 =23451
a=51749
b=0
m=244944
n=1000

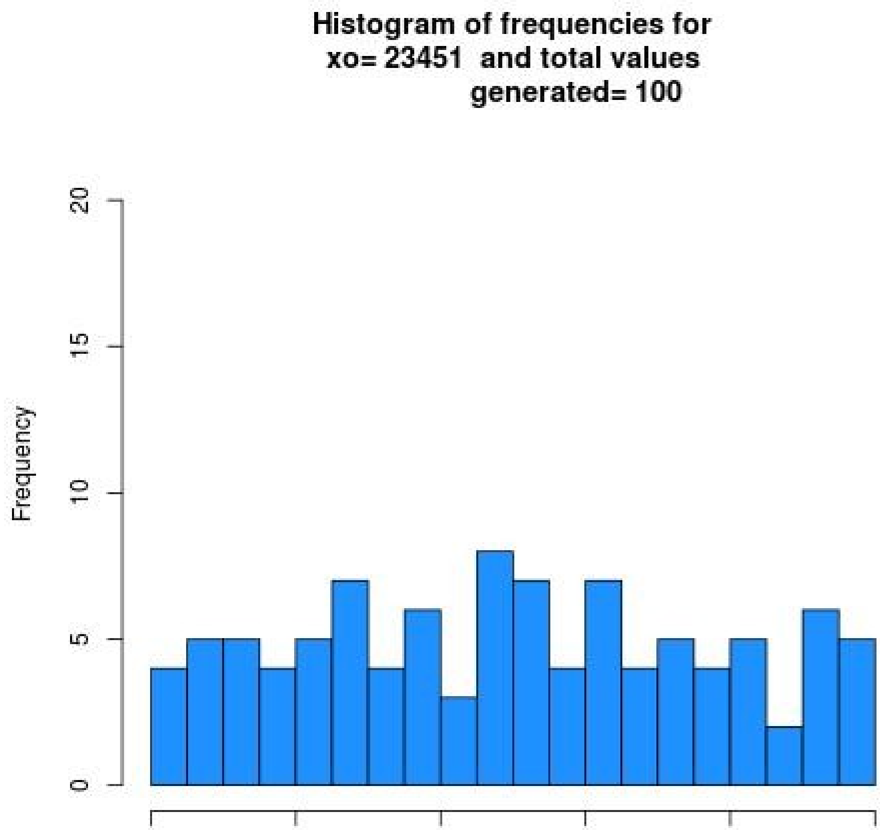


$X_0 = 2341$
a=51749
b=0
m=244944
n=1000

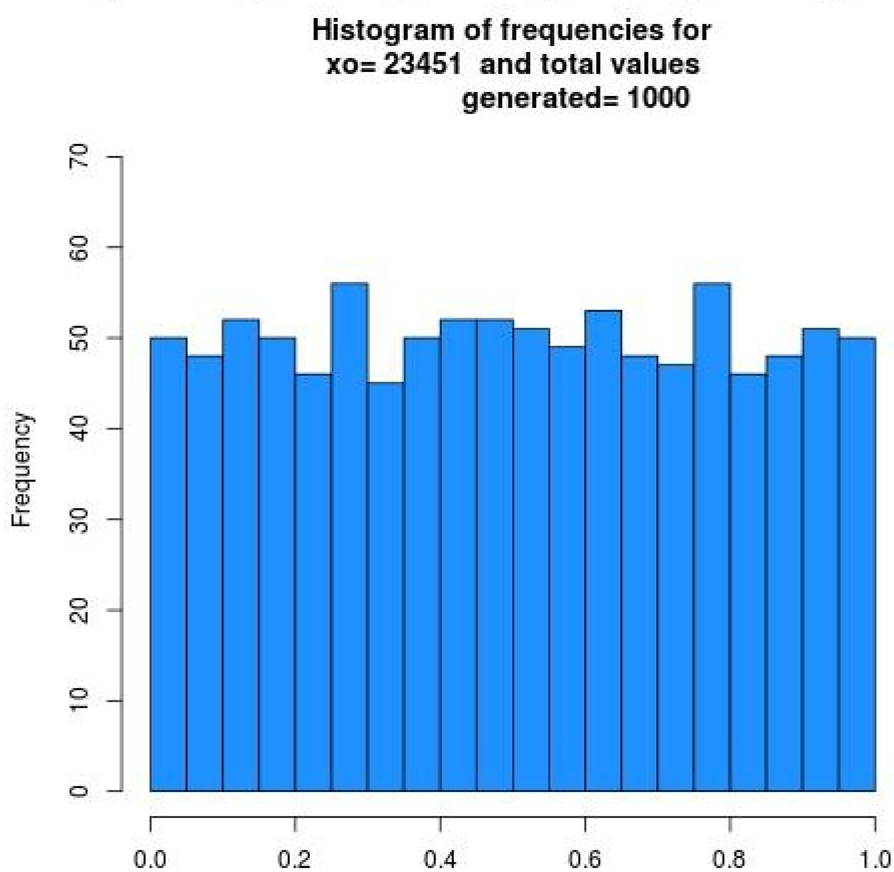


<u>**HISTOGRAMS FOR THE SECOND FREQUENCY TABLE:**</u>

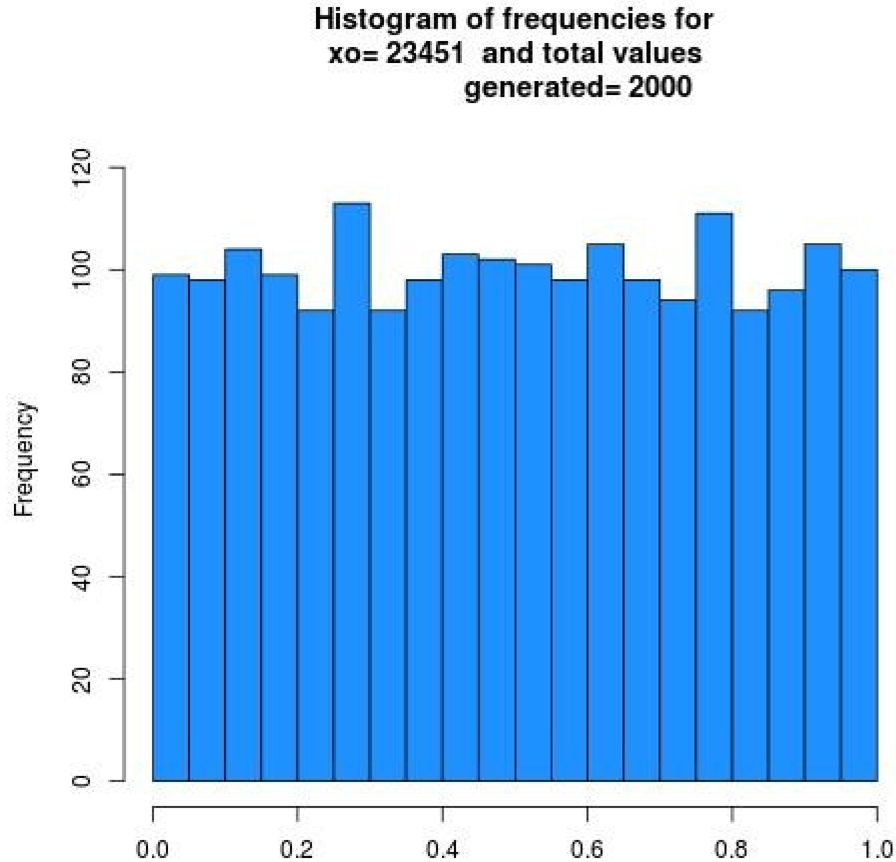
N=100
X_0 =23451
a=51749
b=0
m=244944



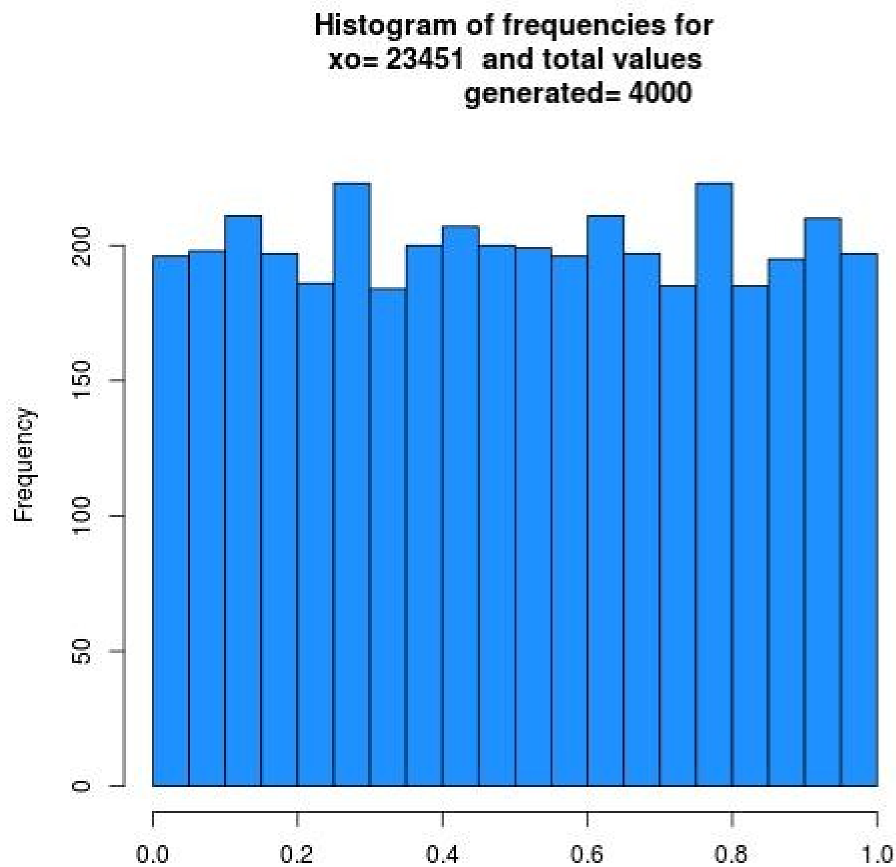
N=1000
X_0 =23451
a=51749
b=0
m=244944



N=2000
X_0 =23451
a=51749
b=0
m=244944

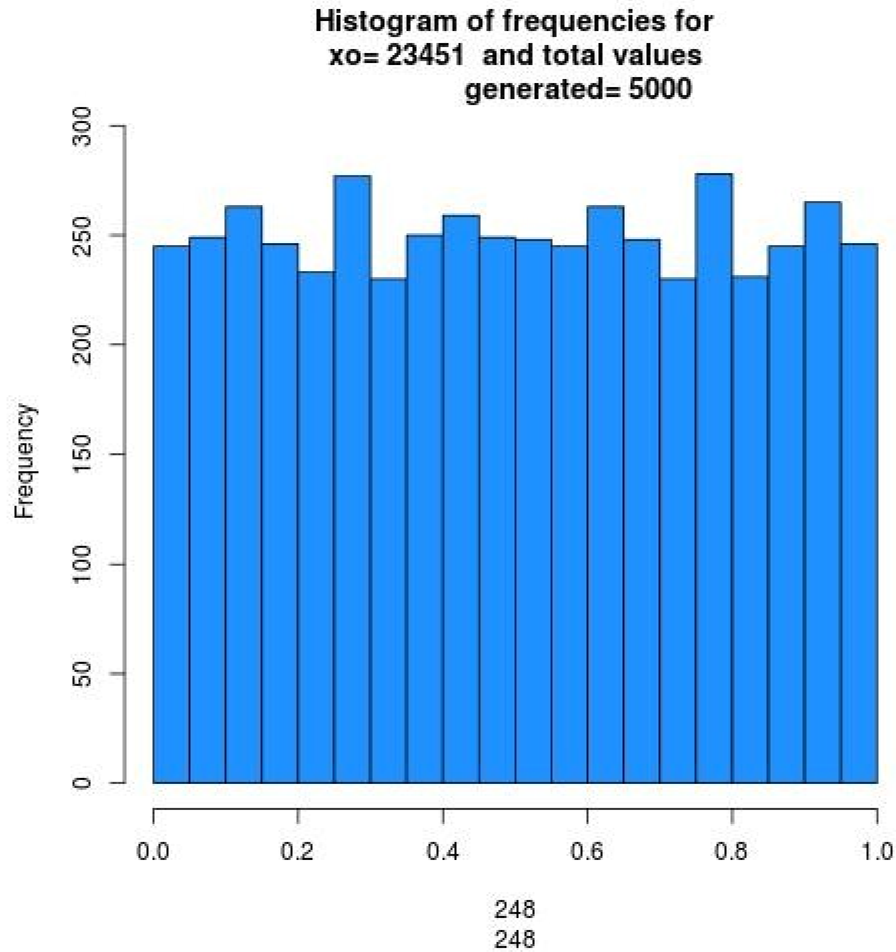


N=4000
X_0 =23451
a=51749
b=0
m=244944



198

N=5000
X_0 =23451
a=51749
b=0
m=244944



OBSERVATIONS

- The seed value x_0 only determines the length of the cycle and as such do not have any effect on the distribution of the random values generated if other parameters remain constant.
- As the number of iterations increase, the distribution becomes more and more uniform and the frequency of each interval becomes almost same.
- The higher the value of 'a' i.e. the multiplying factor, the change between the consecutive values of random number in the sequence is more evident.

RESULT

- The higher the value of 'n' i.e. the number of iterations, more uniform the distribution of the generated random values become.

QUESTION-III

Generate a sequence u_i with $a=1229, b=1, m=2048$. Plot in a two-dimensional graph the points (u_{i-1}, u_i) i.e., the points $(u_1, u_2), (u_2, u_3), (u_3, u_4), \ldots$ What are your observations?

RCODE :

```
func_rand <- function(val, m){
        return(val/m);
}

func_val <- function(val, a, b, m){
        return(((a*val)+b)%%m);
}

func <- function(initial, a, b, m, end){
        count<-1;
        arr<-array(20);
        x<-array();
        y<-array();

        for(i in 1:20)
        {
                arr[i]=0
        }


        val <- initial;
        random <- func_rand(val, m);

        val <- func_val(val, a, b, m);
        random <- func_rand(val, m);
        prev<-random;

        while(count<=end){
                val <- func_val(val, a, b, m);
                random <- func_rand(val, m);

                y[count] = random;
                x[count] = prev;
```

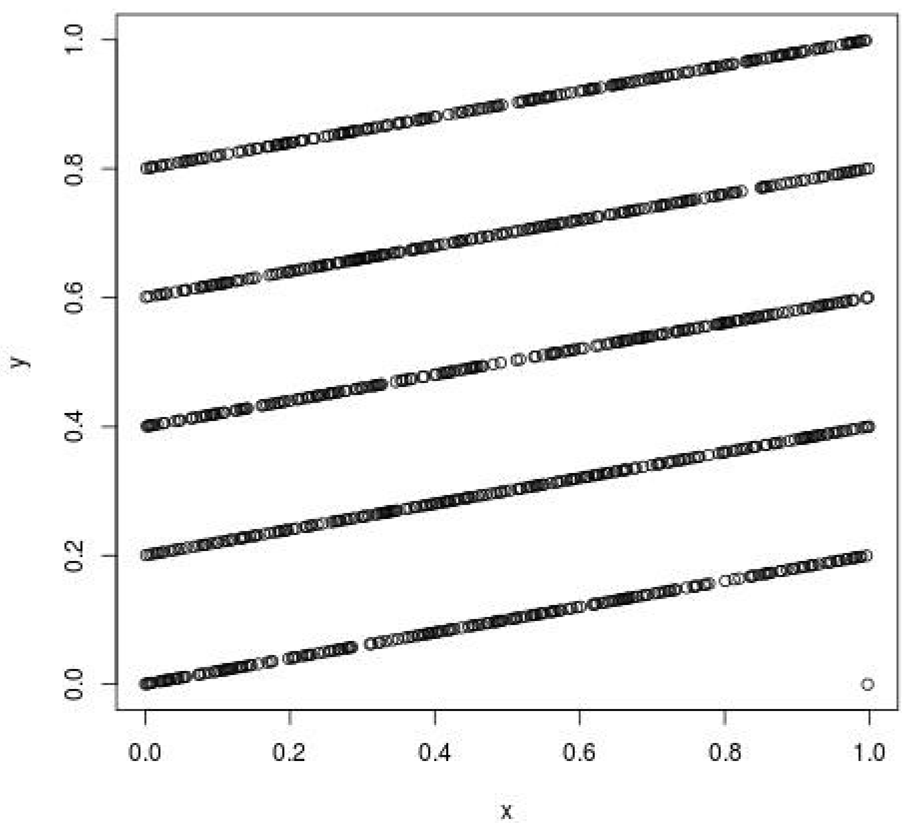
```
            prev = random;
            count <- count+1;
        }

        plot(x,y);
        dev.copy(jpeg,paste("Graph.jpg"));
        dev.off();
}

main <- function(){
        func(172, 1229, 1, 2048, 1000);
}
```

OUTPUT

The plot of $(u_1, u_2), (u_2, u_3), (u_3, u_4), \ldots$ comes to be be as follows:



OBSERVATIONS

- We can clearly see from the plot of $(u_1, u_2), (u_2, u_3), (u_3, u_4), \dots$ that we have got parallel lines separated uniformly for each other(around 0.2 units).
- In RANDU generator, $X_i = 65539X_{i-1} \bmod 2^{31}$ plot we get all the random numbers fall entirely on 15 hyperplanes . And here we got all the random numbers on the straight lines in the 2d plot. This is because 'm' is 2048 in this case which is equal to 2^{11} which was 2^{31} in the case of RANDU Generator.

RESULT

- Due to serial correlation between successive values of the sequence u_n the plot between u_i and u_{i+1} is a set of parallel lines. This is a direct consequence of Marsaglia's theorem which says that of Linear Congruential Generator(LCG) is used to choose points in a n-dimensional space, the points will lie on at most $m^{1/n}$ hyperplanes. That is why in this case we observe a set of parallel lines.
- As the plot obtained consists of parallel lines, this is a not a good function for generating random numbers.