LAB REPORT ON

SIMULATION AND MODELING

SUBMITTED BY : SAMYAK MANANDHAR

SUBMITTED TO : TEKSAN GHARTI MAGAR

FACULTY : BSC.CSIT 5TH SEMESTER

C:\Users\PC\Documents\prime college\Fifth Semester\samyak manandhar\SIMULATION\1.exe

```
Samyak Manandhar 79010513
Enter number of iterations
4
Head=  1
Tail = 3
Tail wins by 2
--------------------------------
Process exited after 3.323 seconds with return value 0
Press any key to continue . . .
```

C:\Users\PC\Documents\prime college\Fifth Semester\samyak manandhar\SIMULATION\2.exe

```
Samyak Manandhar 79010513
Enter the number of iterations used to estimate pi: 3
Value of PI =  1.333333
--------------------------------
Process exited after 5.601 seconds with return value 0
Press any key to continue . . .
```

Select C:\Users\PC\Documents\prime college\Fifth Semester\samyak manandhar\SIMULATION\3.exe

```
Samyak Manandhar 79010513
Number of points =  6331
The area =  2.532400
--------------------------------
Process exited after 0.1728 seconds with return value 0
Press any key to continue . . .
```

C:\Users\PC\Documents\prime college\Fifth Semester\samyak manandhar\SIMULATION\4.exe

```
0.376238
0.336634
0.950495
0.435644
0.415842
0.722772
0.465347
0.455446
0.108911
0.980198
0.475248
0.801980
0.237624
0.485149
0.148515
Samyak Manandhar 79010513

--------------------------------
Process exited after 0.1523 seconds with return value 0
Press any key to continue . . .
```

C:\Users\PC\Documents\prime college\Fifth Semester\samyak manandhar\SIMULATION\5.exe

```
Samyak Manandhar 79010513
How many random numbers to be generated
4
x =  72
   72 x =  17
   17 x =  28
   28 x =  78
   78
--------------------------------
Process exited after 2.051 seconds with return value 0
Press any key to continue . . .
```

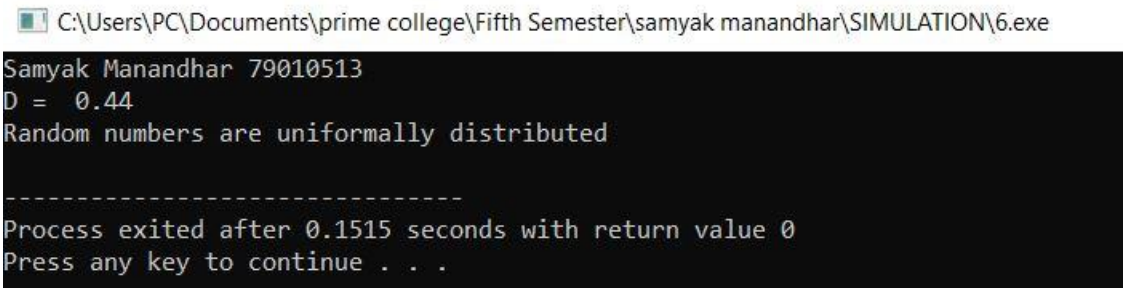
Lab 6: Write a C program to that tests random numbers for frequency using Kolmogorov S test.

Code:

```
#include<conio.h>

#include<stdio.h>

#define N 5

#define Dalpha 0.665

int main(){

        printf("Samyak Manandhar 79010513\n");

        float R[N] = {0.05,0.14,0.44,0.81,0.93};

        float X[N],Y[N],Z[N];

        float D1,D2,D;

        int i;

        for(i=1;i<=N;i++){

                X[i-1] = (double)i/N;

        }

        for(i=1;i<=N;i++){

                Y[i-1] = (double)i/N-R[i-1];

        }

        for(i=1;i<=N;i++){

                Z[i-1] = R[i-1]-(double)(i-1)/N;

        }

        D1 = Y[0];

        for(i=1;i<N;i++)  {

                if(D1<Y[i])

                D1 = Y[i];

        }

        D2 = Z[0];

   for(i=1;i<N;i++){

                if(D1<Z[i])
```

```
            D1 = Z[i];
        }
        D = (D1>D2)?D1:D2;
        printf("D =  %0.2f",D);
        if(D<Dalpha)
        printf("Random numbers are uniformally distributed\n");
        else
        printf("Random numbers are not uniformally distributed\n");

}
```

Output:



Lab 7: Write a C program that tests random numbers for frequency using chi-Square test.

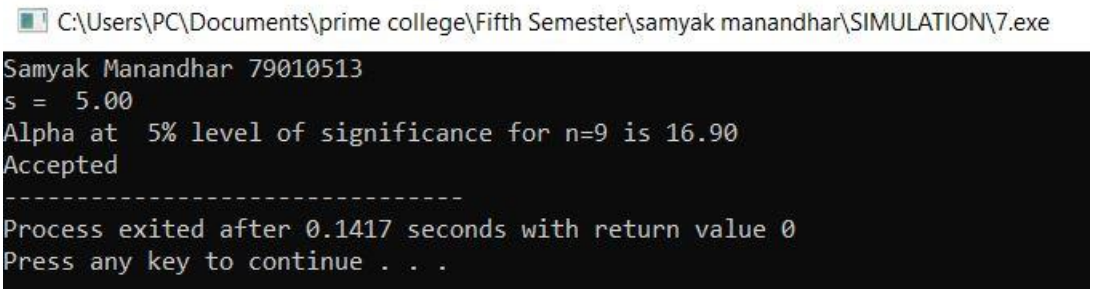
Code:

```
#include<stdio.h>
#include<conio.h>
#define N 100
#define alpha 16.9
void sort(float arr[]);
int main(){
        printf("Samyak Manandhar 79010513\n");
        int i,j;
        float XO[10];
        float XE[10];
        float XOE[10];
        float XOE2[10];
        float R[10];
        float s=0.0;
        float x[] = { 0.34,0.83,0.96,0.47,0.79,0.37,0.99,0.37,0.72,0.06,0.18,0.90,
        0.76,0.99,0.30,0.71,0.17,0.51,0.43,0.39,0.26,0.25,0.79,
        0.77,0.17,0.23,0.99,0.54,0.56,0.84,0.97,0.89,0.64,0.67,
        0.82,0.19,0.46,0.01,0.97,0.24,0.88,0.87,0.70,0.56,0.56,
        0.82,0.05,0.81,0.30,0.40,0.64,0.44,0.81,0.41,0.05,0.93,
        0.66,0.028,0.94,0.64,0.47,0.12,0.94,0.52,0.45,0.65,0.10,
        0.69,0.96,0.40,0.60,0.21,0.74,0.73,0.31,0.37,0.42,0.34,
        0.58,0.19,0.11,0.46,0.22,0.99,0.78,0.39,0.18,0.75,0.73,0.79,
        0.29,0.67,0.74,0.02,0.05,0.42,0.49,0.49,0.05,0.62,0.78 };
        sort(x);
        for(i=0;i<10;i++){
                XO[i]=0.0;
                XE[i]=10.0;
```

```
}
for(i=0;i<N;i++) {
   if(x[i]<=0.1)
            XO[0]++;
            else if(x[i]<=0.2)
            XO[1]++;
            else if(x[i]<=0.3)
            XO[2]++;
            else if(x[i]<=0.4)
            XO[3]++;
            else if(x[i]<=0.5)
            XO[4]++;
            else if(x[i]<=0.6)
            XO[5]++;
            else if(x[i]<=0.7)
            XO[6]++;
            else if(x[i]<=0.8)
            XO[7]++;
            else if(x[i]<=0.9)
            XO[8]++;
            else if(x[i]<=1.0)
            XO[9]++;
}
for(i=0;i<10;i++){
            XOE[i] = XO[i]-XE[i];
            XOE2[i] = XOE[i]*XOE[i];
            R[i] = XOE2[i]/XE[i];
            s =  s+R[i];
}
```

```
        printf("s =  %0.2f\n",s);
        printf("Alpha at  5%% level of significance for n=9 is %0.2f\n",alpha);
        if(s<=alpha)
        printf("Accepted");
        else
        printf("Rejected");
}
void sort(float x[]){
        int i,j;
        float temp;
        for(i=0;i<N;i++){
                for(j=0;j<N-1;j++){
                        if(x[j+1]<x[j]){
                                temp =  x[j];
                                x[j] =  x[j+1];
                                x[j+1] = temp;
                        }
                }
        }
        }
```

Output:

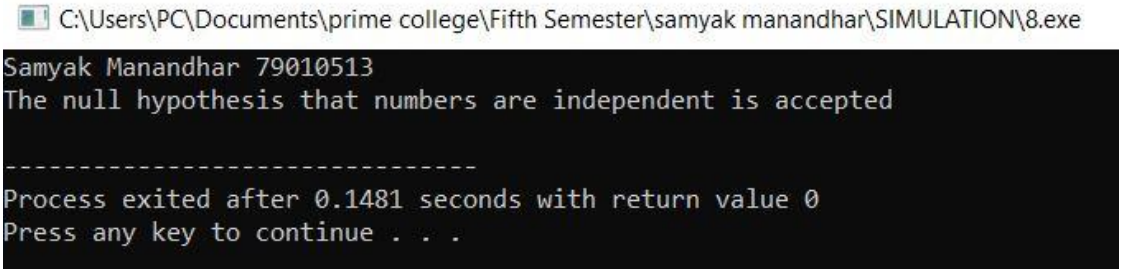


Lab 8: Write a program to test random numbers for independence using autocorrelation method

Code:

```
#define N 30
#include<stdio.h>
#include<math.h>
int main(){
        printf(“Samyak Manandhar 79010513\n”);
        int i,m,M,k;
        i=2,m=5;
        M = ((N-i)/m)-1;
        float s35=0,r35,z0;
        floatR[]={0.12,0.01,0.23,0.28,0.89,0.31,0.64,0.28,0.83,0.93,0.99,0.15,0.33,0.35,0.91,0.41,0.60,0.
27,0.75,0.88,0.68,0.49,0.05,0.43,0.95,0.58,0.19,0.36,0.69,0.87};
        for(k=0;k<=M;k++){
                s35 =s35+R[i+k*m]*R[i+(k+1)*m];}
                s35 = s35/(M+1);
                s35 = s35-0.25;
                r35 = sqrt(13*M+7)/(12*(M+1));
          z0 = r35/s35;
          if(z0<=1.96)
          printf("The null hypothesis that numbers are independent is accepted\n");
          else
        printf("The null hypothesis that numbers are independent is not accepted\n");
        return 0;}
```

Output:



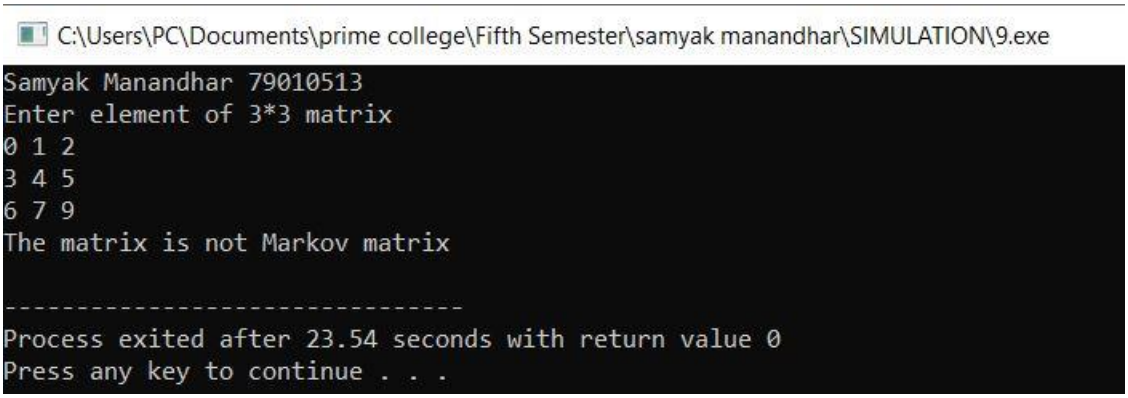
Lab 9: Write a program to test whether the given matrix is Markov or not

Code:

```
#include<stdio.h>
#define N 3
#define M 3
int isMarkovMatrix(float m[][N]){
	printf(“Samyak Manandhar 79010513\n”);
	int i,j,s,t=1;
	for(i=0;i<M;i++){
		s=0;
		for(j=0;j<N;j++){
		   s =  s+m[i][j];
		}
		if(s>1){
			t=0;
			break;
		}
	}return t;
}
void read(float m[][N]){
	printf("Enter element of %d*%d matrix\n",M,N);
	for(int i=0;i<M;i++){
		for(int j=0;j<N;j++){
			scanf("%f",&m[i][j]);
		}
	}
}
int main(){
	float matrix[M][N];
```

```
    read(matrix);
    if(isMarkovMatrix(matrix))
    printf("The matrix is Markov Matrix\n");
    else
    printf("The matrix is not Markov matrix\n");
    return 0;
}
```

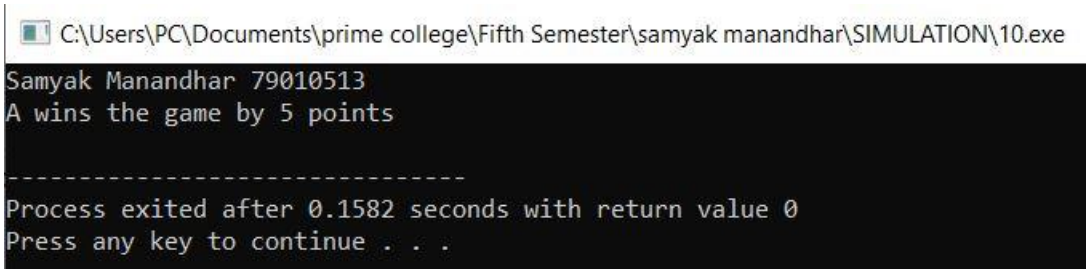
Output:



Lab 10: Write a program to simulate the game called DiceToss.

Code:

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<time.h>
int Arand(){int r;
        r =  1+rand()%5;
        return r;}
int Brand(){int r;
        r =  1+rand()%5;
        return r;}
int main(){
        printf("Samyak Manandhar 79010513\n");
        srand(time(0));
        int sa=0,sb,a,b,x,y;
        a = Arand();
        b = Arand();
        sa =  a+b;
        x =  Brand();
        y =  Brand();
        sb =  x+y;
        if(sa>sb)
        printf("A wins the game by %d points\n",sa);
        else
        printf("B wins the game by %d points\n",sb);
        return 0;
}
```



Lab 11: Write C program to simulate single server queuing system

Code:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<stdlib.h>
int main(){
        printf("Samyak Manandhar 79010513\n");
        int kk,i,j,run=10;
        float x,iat,st, awt, pcu,wt=0,it=0;
        float mean=10.0, sd= 1.5, mue=9.5, sigma=1.0;
        float sb = 0.0,se=0.0,cit=0, cat=0, cwt=0;
        printf("\nIAT    CAT    SB    ST    SE    CWT    CIT");
        for(j=1;j<=run;++j)        {
                float sum=0;
                for(i=1;i<=12;++i){
                        x = rand()/32768.0;
                        sum  =  sum+x;
                }
                        iat =  mean+sd*(sum-6.0);
                        cat =  cat+iat;
                        if(cat<=se){
                                sb = se;
                                wt = se-cat;
                                cwt = cwt+wt;
                        }
                        else {
                                sb = cat;
                                it = sb-se;
```

```
                cit = cit+it;
            }
            sum = 0;
            for(i=1;i<=12;++i){
                x = rand()/32768.0;
                sum = sum+x;
            }st = mue+sigma*(sum-6.0);
                se = sb+st;
                printf("\n %5.2f %6.2f %6.2f %6.2f %6.2f %6.2f %6.2f",iat,cat,sb,st, se,
cwt,cit);
            }
            awt =  cwt/run;
            pcu = ((cat-cit)*100.0)/cat;
            printf("\n Average waiting time =  %6.2f",awt);
            printf("\nPercentage capacity untilization =  %6.2f",pcu);
        }
```

Output:

```
C:\Users\PC\Documents\prime college\Fifth Semester\samyak manandhar\SIMULATION\11.exe
Samyak Manandhar 79010513
IAT     CAT     SB      ST      SE      CWT     CIT
 10.72  10.72  10.72   7.37  18.09   0.00  10.72
  8.49  19.21  19.21  10.77  29.98   0.00  11.84
 11.68  30.90  30.90   9.02  39.92   0.00  12.76
 10.74  41.63  41.63   9.47  51.10   0.00  14.47
  9.09  50.73  51.10   9.90  61.01   0.37  14.47
  9.97  60.70  61.01   9.99  70.99   0.68  14.47
 11.35  72.05  72.05   9.16  81.21   0.68  15.53
 10.74  82.79  82.79   7.08  89.87   0.68  17.11
  9.19  91.98  91.98  10.53 102.51   0.68  19.22
  8.54 100.52 102.51  10.85 113.37   2.67  19.22
 Average waiting time =    0.27
Percentage capacity untilization =   80.88
--------------------------------
Process exited after 0.1377 seconds with return value 0
Press any key to continue . . .
```