Problem 1

$F(x) = x\,,\quad 0 < x < 1$

$$D^+ = \max_{k=1,\dots,N} \left(\frac{k}{N} - F(x_k)\right)$$

$$= \max\{(\frac{1}{3} - F(0.2)), (\frac{2}{3} - F(0.6)), (\frac{3}{3} - F(0.7))\}$$

$$= \max\{\frac{2}{15}, \frac{1}{15}, \frac{3}{10}\}$$

$=0.3$

$$D^- = \max_{k=1,\dots,N} \left(F(x_k) - \frac{k-1}{N}\right)$$

$$= \max\{(F(0.2) - \frac{0}{3}), (F(0.6) - \frac{1}{3}), (F(0.7) - \frac{2}{3})\}$$

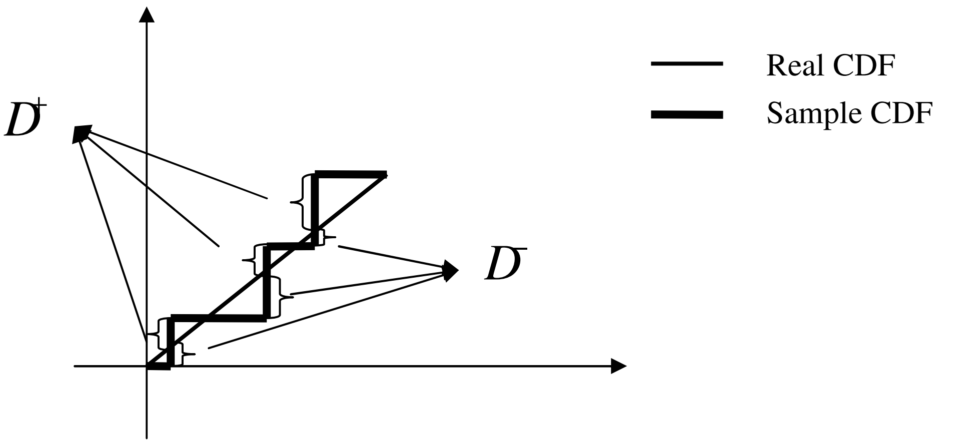
$$= \max\{\frac{2}{10}, \frac{4}{15}, \frac{1}{30}\}$$

$=0.2666$

$$D^N = \max(D^+, D^-) = \max(0.3, 0.2666) = 0.3$$

$\boxed{D^+}$

D^+ is the largest value of how much the sample cumulative distribution function (CDF) over the real cumulative distribution function. It can measure how far the sample away from the real CDF. D^- is the largest value of how much the real CDF over sample CDF. It can measure how below the sample CDF from the real CDF. D_N is the largest difference between the sample CDF and the real CDF.



Problem 2

$$\max\left[F(x_1), \max_{k=1,\dots,N-1}\left(F(x_{k+1})-\frac{k}{N}, \frac{k}{N}-F(x_k)\right), 1-F(x_N)\right]$$

=

$$\max\left[F(x_1), F(x_2)-\frac{1}{N}, \frac{1}{N}-F(x_1), F(x_3)-\frac{2}{N}, \frac{2}{N}-F(x_2), \dots, F(x_N)-\frac{N-1}{N}, \frac{N-1}{N}-F(x_{N-1}), 1-F(x_N)\right]$$

=

$$\max\left[F(x_1)-\frac{0}{N}, F(x_2)-\frac{1}{N}, \frac{1}{N}-F(x_1), F(x_3)-\frac{2}{N}, \frac{2}{N}-F(x_2), \dots, F(x_N)-\frac{N-1}{N}, \frac{N-1}{N}-F(x_{N-1}), \frac{N}{N}-F(x_N)\right]$$

=

$$\max\left[\max_{k=1,\dots,N}\left(\frac{k}{N}-F(x_k)\right), \max_{k=1,\dots,N}\left(F(x_k)-\frac{k-1}{N}\right)\right]$$

(Combine all the $\frac{k}{N}-F(x_k)$ form, and combine all the $F(x_k)-\frac{k-1}{N}$ form.)

Problem 3

Result:

H_0: the sample is from U(0,1)

D_N=0.036956

P-value=0.130285

We assume α=0.05

P-value>α. Therefore do not reject H_0

Algorithm:

1. Write a head file, mc_gen.h, which can generate MCG random numbers.

$$x_n = a * x_{n-1} \bmod M$$

Where $a = 23$, $M = 10^8 + 1$, $x_0 = 47594118$

2.. Write a head file, KStest.h, which can test whether given numbers are random numbers or not. Let the sample be stored in *ivector[i] i=0 to999*

for i=0 to 999

$$ivector2[i] = \max\left[(\frac{i}{N} - ivector(i)), ivector(i) - \frac{i-1}{N}) \right]$$

end for

Then we sort ivector2[i], and the last term (largest term) is the D_N.

Because we have $\sqrt{N}D_N$'s CDF, we can find p-value=1-$F(\sqrt{N}D_N)$

3. Write a source file, which include the main function and print the answer.

Problem 4

We use the RANDU random numbers here. $x_n = a * x_{n-1} \bmod M$

Where $a = 65539$, $M = 2^{31}$, $x_0 = 1$. We know that it is not a good generator, because we can find the 15 planes in 3D.

Result:

We got 10 χ^2 statistics form the RANDU random numbers, which are 14.2, 8.24, 9.5, 12.14, 14.08, 6.04, 8.52, 9.74, 14.8 and 15.

Then we test the 10 χ^2 statistics whether came from χ^2 distributions or not by KS-test.

H_0: the sample is from χ^2 distributions

The p-value=5.5466e-10.

We assume α=0.05

P-value $< \alpha$.

Therefore we reject the H_0.

Algorithm:

We use both C++ and Matlab in this problem.

1. We have the head file, mc_gen.h, in problem 3 which can generate RANDU random numbers by giving $a = 65539$, $M = 2^{31}$, $x_0 = 1$.
2. Write a head file, ChiSquTest.h, which can test whether given numbers are random numbers or not by chi-square goodness-of-fit test.
   We hope to find Q_{k-1} in the ChiSquTest.h. Where
   $$Q_{k-1} = \sum_{i=1}^{k} \frac{(Y_i - np_i)^2}{np_i}$$
   We stored the sample in vector *vec[i],i=0 to999, and we have subinterval k=10.* We find every *vec[i]* belong to which subinterval and count the number of samples in every subinterval, denote Y_j *j=0 to 9*

```
For i= 0 to 999
      For j= 0 to 0.9
            If vec[i] ∈ (j,j+0.1)
                  Y_j = Y_j +1;
            end if
      end for
end for
```

Then we could get Q_{k-1}.

3. We got the 10 Q_{k-1}'s and then we use KS-test to decide whether the 10 Q_{k-1}'s came from χ^2 distributions by Matlab.

```
[H,P,KSSTAT,CV] = kstest(X,cdf,alpha);
```

Where P is the p-value.

Problem 5

Result:

H_0: The Fibonacci generator is from U(0,1)

Q_{k-1}=24.22

P-value=2.269e-5

We assume α=0.05

P-value$<\alpha$. Therefore we reject H_0

Algorithm:

1. Write a head file, fibo_gen.h, which can generate Fibonacci random numbers.

$$x_n = x_{n-1} + x_{n-2} (\bmod M)$$

Where $M = 2^{31}$, $x_0 = 1$, $x_1 = 1$ and we start with the 101st term.

2.. Write a head file, gap_test.h, which can test whether given numbers are random numbers or not. We store the Fibonacci random numbers in *vec[i] i=0 to9999. J=(0.3,0.8).* We store the times of gap i in vector *gap[i]*,i=0 to t-1 and We store the times of gap $\geq$t in *gap[t]*

```
for i=0 to 9999
        if vec[i] ∈ J
                count the gaps and store in gap[i] if gap=i
                                       and in gap[t] if gap≥t
        else
                gap=gap+1;
        end if

        if Σ_{i=1}^{t} gap[i] =100
                stop the loop
        end if

end for
```

3. find $Q_{k-1} = \sum_{i=1}^{k} \frac{(Y_i - np_i)^2}{np_i}$. Where $Y_i = gap[i]$, $i = 0$ to t, n=100,

$p_i = (1-p)^i p$ if i= 0 to t-1

$= (1-p)^i$ if i=t

4. We have Q_{k-1}=24.22 and we can get P-value=2.269e-5 by Matlab.

```
p=1-chi2cdf(Q,k-1)
```

We assume α=0.05 and P-value $<\alpha$. Therefore we reject H_0.

Problem 7

Result:

H_0: The classical MCG random numbers are from U(0,1)

Collision=212

The cumulative probabilities, and the percentage points:

Cumulative probabilities	0.011742	0.057543	0.260926	0.512275	0.756484	0.953438	0.990032	1
Percentage points:	160	168	179	187	195	207	215	233

We assume α=0.05

P-value is between 0.990032 and 1.

Therefore P-value $<\alpha$. We reject H_0

Algorithm:

1. Using the head file, mc_gen.h, in problem 3 which can generate MCG random numbers.

$$x_n = a * x_{n-1} \bmod M$$

Where $a = 23$, $M = 10^8 + 1$, $x_0 = 47594118$

2. Write a head file, coll_test.h, can compute the collisions and the cumulative probabilities, and the percentage points.
   We store the random numbers in *VEC[i]*, i=0 to 1999, and we assume vector urn[i], i=0 to 9999 is the 10000 urns. We compute the collisions first.

```
For i=0 to 1999
        If   a=floor(vec[i]*1000) and urn[a] is not empty
             Collision=collision+1;
        End if
        urn[a]=urn[a]+1;
end   for
```
Output: Collision=212

3. In coll_test.h we also compute the cumulative probabilities, and the percentage points. First, we find the probability matrix [$P_{j,n}$].

$P_{j,n}$ means the P{j urns occupied in n tosses} and we have

$$P_{j,n} = \frac{j}{M} P_{j,n-1} + \frac{M-j+1}{M} P_{j-1,n-1}.$$

Then we have P{collision $\leq$ x }= $\sum_{i=0}^{x} P_{N-i,N}$, after computing we have

Cumulative probabilities	0.011742	0.057543	0.260926	0.512275	0.756484	0.953438	0.990032	1
Percentage points:	160	168	179	187	195	207	215	233

4. We can compare the sample collisions and above table, and then we have P-value is between 0.990032 and 1.We assume α=0.05 Therefore P-value < α. We reject H_0.

Problem 8