```
#define gen_max 1000
#define D 2
#define NP 10
#define F 0.7
#define CR 0.9
#define inibound_l 0.0
#define inibound_h 1.0

/*----Constant for rnd_uni()-----------------------*/
#define IM1 2147483563
#define IM2 2147483399
#define AM  (1.0/IM1)
#define IMM1  (IM1-1)
#define IA1   40014
#define IA2   40692
#define IQ1   53668
#define IQ2   52774
#define IR1   12211
#define IR2   3791
#define NTAB  32
#define NDIV  (1+IMM1/NTAB)
#define EPS1   1.2e-7
#define RNMX   (1.0-EPS1)

#include<stdlib.h>
#include<stdio.h>
#include<time.h>
#include<math.h>
#include<conio.h>

double evaluate(double [],long *);
double evaluate(double tmp[],long *nfe)
 {     double cost;  (*nfe)++;         /* tmp[0]=x1, tmp[1]=y  */

       cost=(-(tmp[1])+(2*tmp[0]*1.4)-log(tmp[0]*1.4/2));
         return(cost);

  } /************** end of evaluate() ******************/

float rnd_uni(long *);
float rnd_uni(long *idum)
       {
      long j; long k;
      static long idum2=123456789;
      static long iy=0;static long iv[NTAB]; float temp;
      if(*idum<=0)
      {
         if(-(*idum)<1)    *idum=1;  else  *idum=-(*idum);   idum2=(*idum);
         for(j=NTAB+7;j>=0;j--)
         {
           k=(*idum)/IQ1;
           *idum=IA1*(*idum-k*IQ1)-k*IR1;
           if(*idum<0)      *idum+=IM1;
           if(j<NTAB)       iv[j]=*idum;
         }
         iy=iv[0];
       }
       k=(*idum)/IQ1;
       *idum=IA1*(*idum-k*IQ1)-k*IR1;
       if(*idum<0)    *idum+=IM1;
       k=idum2/IQ2;
       idum2=IA2*(idum2-k*IQ2)-k*IR2;
       if(idum2<0)         idum2+=IM2;
       j=iy/NDIV;      iy=iv[j]-idum2;   iv[j]=*idum;
       if(iy<1)    iy+=IMM1;
       if((temp=AM*iy)>RNMX)     return RNMX;
       else                      return temp;
     }

void main()

{
   int i,j,k,a,b,c,good,count=0,seed; long nfe=0;
   double x1[NP][D],x2[NP][D],cost[NP],trial[D],cost_trial,pen,lhs1,costmax,costmin;
   clock_t start, end;
 printf("\nseed=");
 scanf("%d",&seed);
 long  rnd_uni_init= -(long)seed;   start = clock();

for (i=0;i<NP;i++)
{

    for (j=0;j<D;j++)
       {  if(j==0)
      x1[i][j]=0.5/1.4 + rnd_uni(&rnd_uni_init)*(1.0-(0.5/1.4));
      else
      x1[i][j]=inibound_l + rnd_uni(&rnd_uni_init)*(inibound_h-inibound_l);
        if(x1[i][1]>=0.5)   x1[i][1]=1.0;   else x1[i][1]=0.0;
       }
      pen=0.0;

    lhs1=((-x1[i][0]*1.4)-log(x1[i][0]*1.4/2)+(x1[i][1]));
    if(lhs1>0.0)
      {
      pen=lhs1*10;
      cost[i]=evaluate(x1[i], &nfe);
      cost[i]=cost[i]+pen;
      continue;
      }

    if(lhs1<=0.0)
       cost[i]=evaluate(x1[i], &nfe);

}
    costmin=cost[0];
      for(i=1;i<NP;i++)
       {  if(costmin>cost[i])
      costmin=cost[i];
       }

while (count<gen_max)
 {
       for (i=0;i<NP;i++)
    {
        do a=int ((rnd_uni(&rnd_uni_init))*NP); while (a==i);

        do b=int (rnd_uni(&rnd_uni_init)*NP); while (b==i || b==a);

        do c=int (rnd_uni(&rnd_uni_init)*NP); while (c==i || c==a || c==b);

        j=int (rnd_uni(&rnd_uni_init)*D);

        for (k=1;k<=D;k++)
          {
           if(rnd_uni(&rnd_uni_init)<CR  || k==D)
              {
                  trial[j]=x1[c][j]+F*(x1[a][j]-x1[b][j]);
              }
             else trial[j]=x1[i][j];

         if(trial[0]<(0.5/1.4))   trial[0]=(0.5/1.4);
         if(trial[0]>1.0)         trial[0]=1.0;
          if(trial[1]>=0.5) trial[1]=1.0;   else trial[1]=0.0;
              j=(j+1)%D;
          }

     pen=0.0;

    lhs1=(-trial[0]*1.4-log(trial[0]*1.4/2)+(trial[1]));
    if(lhs1>0.0)
      {
      pen=lhs1*10;
      cost_trial=evaluate(trial, &nfe);
      cost_trial=cost_trial+pen;

      if(cost_trial<=cost[i])
        {
            for (j=0;j<D;j++)
            x2[i][j]=trial[j];
            cost[i]=cost_trial;
          if(cost_trial<costmin)
           {
            costmin=cost_trial;
              /* imin=i;
            assignd(best,trial); */
           }
        }
        else for (j=0;j<D;j++)
            x2[i][j]=x1[i][j];
     continue;
      }
    if(lhs1<=0.0)
     {
    cost_trial=evaluate(trial, &nfe);

      if(cost_trial<=cost[i])
        {
            for (j=0;j<D;j++)
            x2[i][j]=trial[j];
            cost[i]=cost_trial;
          if(cost_trial<costmin)
           {
            costmin=cost_trial;
              /* imin=i;
            assignd(best,trial); */
           }
        }
        else for (j=0;j<D;j++)
            x2[i][j]=x1[i][j];
    continue;
     }
   } /************* end of for loop ****************/

      for (i=0;i<NP;i++)
    {
        for (j=0;j<D;j++)
        x1[i][j]=x2[i][j];
    }

    costmax=cost[0];
      for(i=1;i<NP;i++)
       {  if(costmax<cost[i])
      costmax=cost[i];
       }
    costmin=cost[0];
      for(i=1;i<NP;i++)
       {  if(costmin>cost[i])
      costmin=cost[i];
       }


   if((costmax-costmin)<0.00001)
       break;
    count++;

}  /*************  end of while loop *******************/

        end = clock();
for(i=0;i<NP;i++)
 {
   for(j=0;j<D;j++)
   if(j==0)
   printf("u[%d]=%lf",j,x1[i][j]*1.4);
   else printf("u[%d]=%lf",j,(x1[i][j]));
   printf("cost[%d]=%lf    ",i,cost[i]);
 }
 printf("NFE=%ld\n",nfe);
 printf("The time was: %f\n", (end - start) / CLK_TCK);
 printf("lhs1=%lf\n",lhs1);
} /************* end of main() ****************/
```