```
/*Code for Thermal Cracker Simulation*/
#define gen_max 1500
#define D 4
#define NP 40
#define F 0.5       /* F=0.5 & CR=0.8*/
#define CR 0.8
#define inibound_l 0
#define inibound_h 90000

/*----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>
#include<memory.h>

void assignd(double a[], double b[]);
void assignd(double a[], double b[])
   {
    int j;
    for(j=0;j<D;j++)
    {
    a[j]=b[j];
    }
   }

float evaluate(double [], double pen, long *);
float evaluate(double tmp[], double pen, long *nfe)
    {     double cost;
          (*nfe)++;
  cost=(tmp[0]*9.1)+(tmp[1]*16.92/9.0)-(tmp[2]*23.2911/9)+(tmp[3]*17.8974/9)-pen;
          return cost;
    }

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,d,e,good,count=0,seed,strategy=7,imin;  long nfe=0;
   double x1[100][10],x2[100][10],cost[100],lhs1,lhs2,lhs3,trial[10];
   double cost_trial,pen,costmin,costmax,bestit[D],best[D];
   clock_t start, end;    FILE *fout;  char ch;

 printf("\nseed=");         scanf("%d",&seed);
/* printf("\nstrategy=");       scanf("%d",&strategy);*/

 long  rnd_uni_init= -(long)seed;     start = clock();

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

    for (j=0;j<D;j++)
       {
      x1[i][j]=inibound_l + rnd_uni(&rnd_uni_init)*(inibound_h-inibound_l);
       }
   pen=0.0;

lhs1=(16.5*x1[i][0])+(10.1*x1[i][1])+(8.861*x1[i][2])+(9.926*x1[i][3]);

   if(lhs1>1800000.0)
    {
     pen=(lhs1*100);
     cost[i]= evaluate(x1[i], pen, &nfe);
     continue;
    }

lhs2=(7.5*x1[i][0])+(4.0*x1[i][1])+(2.14*x1[i][2])+(2.665*x1[i][3]);
   if(lhs2>450000.0)
   {
     pen=(lhs2*100);
     cost[i]=evaluate(x1[i],pen,&nfe);
     continue;
   }

lhs3=(0.15*x1[i][0])+(1.51*x1[i][1])+(1.3711*x1[i][2])+(1.6426*x1[i][3]);
   if(lhs3>180000.0)
   {     pen=(lhs3*100);
     cost[i]=evaluate(x1[i],pen,&nfe);
     continue;
   }

  if(lhs1<=1800000 && lhs2<=450000.0 && lhs3<=180000.0)
    {
       cost[i]=evaluate(x1[i],pen,&nfe);
    }
}

   costmax=cost[0];
    imin=0;
       for(i=1;i<NP;i++)
    {
        if(cost[i]>costmax)
         {
        costmax=cost[i];
        imin=i;
         }
    }
    assignd(best,x1[imin]);
    assignd(bestit,x1[imin]);


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);

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

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

  /*------------------de/rand/1/bin---------------------------*/
        if (strategy==1)
         {
        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[j]<0.0)        trial[j]=0.0;

        j=(j+1)%D;

         }
          }
/*----------------------------DE/best/1/bin-------------------------*/

 else if (strategy==2)
     {
            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]=bestit[j]+F*(x1[a][j]-x1[b][j]);
             }
             else trial[j]=x1[i][j];

             if(trial[j]<0.0)       trial[j]=0.0;

           j=(j+1)%D;
        }
      }

/*------------------de/best/2/bin------------------------*/
  else if (strategy==3)
      {
            assignd(trial,x1[i]);

            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]=bestit[j]+F*(x1[a][j]+x1[b][j]-x1[c][j]-x1[d][j]);
            }
            else trial[j]=x1[i][j];
           if(trial[j]<0.0)     trial[j]=0.0;

           j=(j+1)%D;
        }
      }

/*------------------de/rand/2/bin------------------------*/
 else if (strategy==4)
     {
            assignd(trial,x1[i]);
            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[e][j]+F*(x1[a][j]+x1[b][j]-x1[c][j]-x1[d][j]);
            }
            else trial[j]=x1[i][j];
          if(trial[j]<0.0)      trial[j]=0.0;

           j=(j+1)%D;
        }
     }

/*------------------de/rand-to-best/1/bin----------------------*/

  else if (strategy==5)
     {
               assignd(trial,x1[i]);

            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]=trial[j]+F*(bestit[j]-trial[j])+F*(x1[a][j]-x1[b][j]);
            }
            else trial[j]=x1[i][j];
           if(trial[j]<0.0)     trial[j]=0.0;

           j=(j+1)%D;
        }
      }
/*------------------de/rand/1/exp----------------------*/

   else if (strategy==6)
     {
        j=int (rnd_uni(&rnd_uni_init)*D);
        k=0;
        do
           {
             trial[j]=x1[c][j]+F*(x1[a][j]-x1[b][j]);
             if(trial[j]<0.0)       trial[j]=0.0;
             j=(j+1)%D;
             k++;
           }
         while((rnd_uni(&rnd_uni_init))<CR && k<D);
     }

 /*------------------de/best/1/exp----------------------*/

else if (strategy==7)
     {
        j=int (rnd_uni(&rnd_uni_init)*D);
        k=0;
        do
           {
             trial[j]=bestit[j]+F*(x1[a][j]-x1[b][j]);
             if(trial[j]<0.0)       trial[j]=0.0;
             j=(j+1)%D;
             k++;
           }
           while((rnd_uni(&rnd_uni_init))<CR && k<D);
     }

/*------------------de/best/2/exp----------------------*/

 else if (strategy==8)
      {

        assignd(trial,x1[i]);
        j=int (rnd_uni(&rnd_uni_init)*D);
        k=0;
       do
          {
        trial[j]=bestit[j]+F*(x1[a][j]+x1[b][j]-x1[c][j]-x1[d][j]);
        if(trial[j]<0.0)        trial[j]=0.0;
```

```
            j=(j+1)%D;
            k++;
         }
         while((rnd_uni(&rnd_uni_init))<CR && k<D);
     }

/*-------------------de/rand/2/exp-----------------------*/

else if (strategy==9)
     {

        assignd(trial,x1[i]);
        j=int (rnd_uni(&rnd_uni_init)*D);
        k=0;
        do
           {
        trial[j]=x1[e][j]+F*(x1[a][j]+x1[b][j]-x1[c][j]-x1[d][j]);
        if(trial[j]<0.0)        trial[j]=0.0;
            j=(j+1)%D;
            k++;
           }
        while((rnd_uni(&rnd_uni_init))<CR && k<D);
     }

/*-------------------de/rand-to-best/1/exp-----------------------*/

  else if(strategy==10)

     {
         assignd(trial,x1[i]);
         j=int (rnd_uni(&rnd_uni_init)*D);
         k=0;
        do
         {
           trial[j]=trial[j]+F*(bestit[j]-trial[j])+F*(x1[a][j]-x1[b][j]);
        if(trial[j]<0.0)        trial[j]=0.0;
            j=(j+1)%D;
            k++;
         }
        while((rnd_uni(&rnd_uni_init))<CR && k<D);
     }
   pen=0.0;

lhs1=(16.5*trial[0])+(10.1*trial[1])+(8.861*trial[2])+(9.926*trial[3]);
  if(lhs1>1800000.0)
    {
     pen=(lhs1*10);
     cost_trial=evaluate(trial,pen,&nfe);
     if (cost_trial>=cost[i])
        {
           for (j=0;j<D;j++)
           x2[i][j]=trial[j];
           cost[i]=cost_trial;
        if(cost_trial>costmax)
           {
           costmax=cost_trial;
           imin=i;
           assignd(best,trial);
           }
        }
        else for (j=0;j<D;j++)
           x2[i][j]=x1[i][j];

    continue;
     }

lhs2=(7.5*trial[0])+(4.0*trial[1])+(2.14*trial[2])+(2.665*trial[3]);
   if(lhs2>450000.0)
    {
     pen=(lhs2*10);
      cost_trial=evaluate(trial,pen,&nfe);
      if (cost_trial>=cost[i])
        {
           for (j=0;j<D;j++)
           x2[i][j]=trial[j];
           cost[i]=cost_trial;
        if(cost_trial>costmax)
            {
             costmax=cost_trial;
             imin=i;
             assignd(best,trial);
            }
        }
        else for (j=0;j<D;j++)
            x2[i][j]=x1[i][j];
     continue;
    }

lhs3=(0.15*trial[0])+(1.51*trial[1])+(1.3711*trial[2])+(1.6426*trial[3]);
    if(lhs3>180000.0)
    {
     pen=(lhs3*10);
      cost_trial=evaluate(trial,pen,&nfe);
     if (cost_trial>=cost[i])
        {
            for (j=0;j<D;j++)
            x2[i][j]=trial[j];
            cost[i]=cost_trial;
        if(cost_trial>costmax)
            {
             costmax=cost_trial;
             imin=i;
             assignd(best,trial);
            }
        }
        else for (j=0;j<D;j++)
            x2[i][j]=x1[i][j];
     continue;
    }

if(lhs1<=1800000 && lhs2<=450000.0 && lhs3<=(180000.0))
    {
      cost_trial=evaluate(trial,pen,&nfe);

      if(cost_trial>=cost[i])
        {
            for (j=0;j<D;j++)
            x2[i][j]=trial[j];
            cost[i]=cost_trial;
        if(cost_trial>costmax)
            {
             costmax=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************/
    assignd(bestit,best);
    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)<0.000001)
       break;
    count++;
} /**********end of while loop***********/

  end = clock();

 for(i=0;i<NP;i++)
   {
    printf("x1=%f   x2=%f  x3=%f    x4=%f   ",x1[i][0],x1[i][1],x1[i][2],x1[i][3]);
    printf("cost[%d]=%f    ",i,cost[i]);
   }

 printf("lhs1=%f    lhs2=%f  lhs3=%f    \n ",lhs1,lhs2,lhs3);
 printf("NFE=%ld\n",nfe);
 printf("The time was: %f\n", (end - start) / CLK_TCK);
 printf("would you like to exit? press Y for exit");
 ch=getch();
 if(ch=='y'||ch=='Y')  { printf("exited"); exit(1);}

  fout=fopen("\\out100.xls","a+");
 /* fprintf(fout,"\nThe out put is\n:");
  for(i=0;i<NP;i++)
   { if(i%10==0)
      {
    fprintf(fout, "x1=%f   x2=%f  x3=%f    x4=%f   ",x1[i][0],x1[i][1],x1[i][2],x1[i][3]);
    fprintf(fout, "cost[%d]=%f \n",i,cost[i]);
      }
   } */
fprintf(fout, "%d   %ld ",strategy,nfe);
fprintf(fout, "%f   ", (end - start) / CLK_TCK);
fprintf(fout, "%d   %d  %f  %f\n",count,seed,F,CR);
 fclose(fout);
} /************* end of main() ***************/
```