```
#include<stdio.h>
#include<conio.h>
#include<alloc.h>
int peg[10],peg1[10];
int count[10];
main()
{
 void cnt(int,int,int []);
 void getdata(int *, int *,int [],int[]);
 int  movetopeg(int,int,int,int[],int [],int);
 int  incmoves(int,int);
 int  verify(int [],int [],int);
 int  ipformat[20]={4,2,4,3,1,1};
 int  opformat[20]={1,1,1,1,1,1};
 int dup(int,int,int[],int);
 int noofpegs=4;
 int noofdiscs=6;
 int done =1;
 int j=0,i,k=0,l,m,n,o,p,temp;
 int moves=0;
 clrscr();
 cnt(noofpegs,noofdiscs,ipformat);
 done= verify(ipformat,opformat,noofdiscs);
 while(done!=0)
 {   k=-1;
     i=0;
     while(i<noofpegs)
     {
       k= peg[i]; /* k contains radius of topmost disc */
       if(k!=-1)
       {
       k= peg[i]; /* k contains radius of topmost disc */
       if(k!=-1)
       {

          peg[opformat[k]-1];
          if(k<m)
          {
             j=peg1[k];
             o=dup(j,k,peg1,noofdiscs);
             if(o==-1) o=k;
             for(n=o+1;n<m;n++)
             {
                j=peg1[n];
                temp=-1;
                if(((opformat[n]==peg1[m]) && (peg1[n]!=opformat[n]))&&(k!
=-1)&&n==peg[peg1[n]-1])
                {
                   temp=n;
                   l=peg1[m];
                }
                if((temp!=-1)  && (j!=l))
                   moves =
movetopeg(j,temp,l,ipformat,opformat,noofdiscs);
```

```
            }
          }/*end of k<m */
          else if((k==m) && (m!=-1))
          {
             temp=-1;
             for(n=0;n<=k-1;n++)
             {
                o=dup(peg1[n],n,peg1,noofdiscs);
                if((opformat[n]==opformat[k])  && (peg1[n] !=opformat[n])
&&(n>o))
                temp=n;
             }
             if(temp!=-1)
             {
                j=peg1[temp];
                for(l=0;l<noofpegs;l++)
                {
                   o=dup(j,temp,peg1,noofdiscs);
                   if((m>temp) &&  (o==-1))
                   {
                   moves
=movetopeg(j,temp,peg1[m],ipformat,opformat,noofdiscs);
                   l=noofpegs;
                   }
                   else if((peg[l] > temp)  &&(j!=l+1) &&(o==-1))
                   {
                   moves =
movetopeg(j,temp,l+1,ipformat,opformat,noofdiscs);
                   temp=peg[j-1];
                   l=noofpegs;
                   }
                   else
                   {
                    if((peg[l]==-1) &&(j!=l+1))
                    {
                      moves = movetopeg(j,peg[j-
1],l+1,ipformat,opformat,noofdiscs);
                     l=noofpegs;
                    }
                   }
                }
            }
            else
            {
            for(n=k+1;n<o;n++)
            {
              j=peg1[k];
              if((k==peg[j-1]) && (k<n))
              {
                moves  =
movetopeg(j,k,peg1[n],ipformat,opformat,noofdiscs);
                o=peg[n-1];
                k++;
              }
```

```
            }
            }

        }/*   end of k==m */
        else if(m==-1)
        {
          j=peg1[k];
          o=dup(j,k,peg1,noofdiscs);
          if(o==-1)o=k;
          temp=-1;
          for(n=k+1;n<=o;n++)
          {
             j=peg1[n];
             if(n==peg[j-1])
             {
              temp=n;
              j=peg1[temp];
             }
             if((temp!=-1) &&(n==o))
                moves =
movetopeg(j,temp,opformat[n],ipformat,opformat,noofdiscs);
             else  if((temp==-1) && (k<o))
             {
            for(l=0;l<noofpegs;l++)
             {
               j=peg1[k];
               if((peg[l]==-1) && (opformat[k]-1!=l))
               {
                moves=movetopeg(j,k,l+1,ipformat,opformat,noofdiscs);
                 l=noofpegs;
                }
            } /* end of else */
            n=peg[j-1]-1;
             } /* end of for */
          }/*end of */
      }/* end of m==-1*/
    }/* end of k!=-1*/
       i++;
 }  /*end of while for no of discs */
 done= verify(peg1,opformat,noofdiscs);/* end of while for done  */
}/*end of while for done */

   printf("\n No: of moves =%d",moves);
}

int movetopeg(int a,int b,int c,int ipformat[],int opformat[],int
noofdiscs)
{
      int mov;
       printf("radius of disc is =%d",b);
      mov = incmoves(a-1,c-1);
      peg1[b] = c;
      if(peg[a-1]!=-1)
      peg[a-1]=dup(a,peg[a-1],peg1,noofdiscs);
```

```
      printf("radius of disc after shifting is =%d",peg[a-1]);
      return mov;
}

int verify(int format[],int opformat[],int noofdiscs)
{
int i, done=0;
for(i=0;i<noofdiscs;i++)
{
  if(format[i]!=opformat[i])
  {
      printf(" \nNot Correct=%d %d",i+1,peg1[i]);
       done = 1;
  }
}
  return done;

}

int  incmoves( int i,int m)
{
         static int moves=0;
         printf("\n%d %d",i+1,m+1);
         peg[m]=peg[i];
         count[m]++;
         moves++;
         if(count[i]>0)
         {
           count[i]--;
           if(count[i] == 0)
            peg[i]=-1;
         }
         return moves;
 }

int dup(int i,int n,int format[ ],int noofdiscs)
{
   int j;
    for(j=0;j<noofdiscs;j++)
    {

       if((i==format[j])  &&(j!=n))  /* n is radius with which comparison
is made */
            return j;

    }
     return -1;
 }



 void getdata(int *pegs,int *discs, int format[], int format1[])
 {
 int i;
```

```
 while(*pegs<3 || (*pegs>5 ) )
 {
 printf("Enter No of pegs( 3<+noofpegs<=5)=");
 scanf("%d",&i);
 *pegs =i;
 }
 while(*discs<=1 || (*discs>8 ))  /* tO SATISFY CONSTRAINTS */
 {
 printf("\nEnter No of discs=");
 scanf("%d",&i);
 *discs=i;
 }
 for (i=0;i<*discs;i++)
 {
   printf("Enter ipformat[%d]=",i) ;
   scanf("%d",&format[i]);
   printf("Enter opformat[%d]=",i);
   scanf("%d",&format1[i]);

 }
 return;
} /* end of getdata*/

void cnt(int noofpegs,int nodiscs,int ipformat[])
{
    int j;
    for(j=0;j<noofpegs;j++)
    {
      peg[j]=-1;
      count[j]=0;
    }
    for(j=nodiscs-1;j>=0;j--)
    {
      count[ipformat[j]-1]++;
      peg1[j]=ipformat[j];
      peg[ipformat[j]-1]=j;

   }/* end of for j*/
}
```