```
/*A8: Mini Project □  Centepede Collisions*/
/*M.GAYATHRI-185001050-CSE-A*/
```

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<string.h>
int N,head[100][2],tail[100][2],length[100],direction[100][2];
int board[30][30];
char direc[100];
int array1[30]={0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,2,2};
int array2[30]={0,1,2,3,4,5,6,7,8,9,0,1,2,3,4,5,6,7,8,9,0,1,2,3,4,5,6,7,8,9};
void direction_set();
void divide(int pos,int x,int y);
void search();
void main()
{
    int i,j;
    memset(board,-1,sizeof(board));
    memset(length,-1,sizeof(length));
    memset(head,-1,sizeof(head));
    memset(tail,-1,sizeof(tail));
    memset(direction,-1,sizeof(direction));
    printf("Enter number of Centipedes : \n");
    scanf(" %d",&N);
    printf("Enter Direction of movement, length of centipede , x co-ordinates and y co-ordinates \n");
    for(i=0;i<N;i++)
    {
        scanf(" %c %d %d %d",&direc[i],&length[i],&head[i][0],&head[i][1]);
    }
    direction_set();
    search();
    printf("  ");
    for(i=0;i<30;i++)
        printf(" %d",array1[i]);
    printf("\n");
    printf("  ");
    for(i=0;i<30;i++)
        printf(" %d",array2[i]);
    printf("\n");
    for(i=0;i<30;i++)
    {
        printf("%02d",29-i);
        for(j=0;j<30;j++)
        {
            if(board[j][29-i]==-5)
            {
                printf(" x");
            }
            else
            {
                printf(" .");
            }

        }
        printf("\n");
    }
    printf("\n");
}
void direction_set()
{
    int i;
    for(i=0;i<N;++i)
    {
        if(direc[i]=='R')
        {
            direction[i][0]=1;
            direction[i][1]=0;
            tail[i][0]=head[i][0]-length[i]+1;
            tail[i][1]=head[i][1];
        }
        else if(direc[i]=='L')
        {
            direction[i][0]=-1;
            direction[i][1]=0;
            tail[i][0]=head[i][0]+length[i]-1;
            tail[i][1]=head[i][1];
        }
        else if(direc[i]=='U')
        {
            direction[i][0]=0;
            direction[i][1]=1;
            tail[i][0]=head[i][0];
            tail[i][1]=head[i][1]-length[i]+1;
        }
        else
        {
            direction[i][0]=0;
            direction[i][1]=-1;
            tail[i][0]=head[i][0];
            tail[i][1]=head[i][1]+length[i]-1;
        }
    }
    return;
}
void divide(int pos,int x,int y)
{
    int length1,length2;
    if(length1=direction[pos][0]!=0)
    {
        abs(head[pos][0]-x);
    }
    else
    {
        abs(head[pos][1]-y);
    }
    length2=length[pos]-1-length1;
    if(length2>0)
    {
        length[N]=length2;
        head[N][0]=x-direction[pos][0];
        head[N][1]=y-direction[pos][1];
        head[N][0]=direction[pos][0];
        head[N][1]=direction[pos][1];
        ++N;
    }
    return;
}
void search()
{
    int flag=1,ifcollision=0;
    int i,j,x,y,p1,p2,m;
    while(flag)
    {
        flag=0;
        for(i=0;i<N;i++)
        {
            if(length[i]>0)
            {
                p1=direction[i][0]!=0?0:1;
                p2=p1==0?1:0;
                if(head[i][p1]+direction[i][p1]<=0 || head[i][p1]+direction[i][p1]>=29)
                {
                    length[i]--;
                }
                else if(board[head[i][0]+direction[i][0]][head[i][1]+direction[i][1]]==-5)
                {
                    length[i]--;
                }
                else
                {
                    ifcollision=0;
                for(j=0;j<N;j++)
                {
                    if(j!=i&&length[j]!=0)
                    {
                        if((head[i][p1]+direction[i][p1]==head[j][p1])||(head[i][p1]+direction[i][p1]==tail[j][p1]))
                        {
                            if((head[j][p2]-head[i][p2])*(head[j][p2]-direction[j][p2]*(length[j]-1)-head[i][p2])<=0)//To prevent checking the collision in the opposite direction
                            {
                                ifcollision=1;
                                length[i]--;
                                x=head[i][0]+direction[i][0];
                                y=head[i][1]+direction[i][1];
```

```
                  if(board[x+1][y]==-5)
                  {
                     board[x][y]=-1;
                     divide(j,x+1,y);
                     break;
                  }
                  else if(board[x-1][y]==-5)
                  {
                     board[x][y]=-1;
                     divide(j,x-1,y);
                     break;
                  }
                  else if(board[x][y+1]==-5)
                  {
                     board[x][y]=-1;
                     divide(j,x,y+1);
                     break;
                  }
                  else if(board[x][y-1]==-5)
                  {
                     board[x][y]=-1;
                     divide(j,x,y-1);
                     break;
                  }
                  else
                  {
                     board[x][y]=-5;
                     divide(j,x,y);
                     break;
                  }

               }
            }
         }
         tail[j][p1]=0;//To make sure the tail is constant in the i loop
      }
      if(ifcollision==0)
      {
         head[i][p1]+=direction[i][p1];
         tail[j][p1]+=direction[j][p1];


      }
      }
   }
}
for(m=0;m<N;m++)
{
   if(length[m]!=0)
   {
      flag=1;
      break;
   }
}
```

```
    }
    return;
}





/*OUTPUT:


Enter number of Centipedes :
2
Enter Direction of movement, length of centipede , x co-ordinates and y co-ordinates
R 5 10 10
D 5 12 12
   0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
29 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
28 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
27 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
26 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
25 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
24 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
23 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
22 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
21 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
20 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
19 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
18 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
17 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
16 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
15 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
14 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
13 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
12 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
11 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
10 . . . . . . . . . . . . x . . . . . . . . . . . . . . . . .
09 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
08 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
07 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
06 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
05 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
04 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
03 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
02 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
01 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
00 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .*/
```