```
#include <stdio.h>
#include <assert.h>
#include <unistd.h>

int size;
int map[30][30],good[30][30],bad[30][30];
int good_steps,bad_steps,steps,meet_I,meet_J;
FILE *fout;

void read_input()
{
        FILE *fp;
        int I;
        int K;

        fp=fopen("INPUT1.TXT","rt");

        assert(fp!=NULL);

        fscanf(fp,"%d",&size);

        for(I=0;I<size;I++)
                for(K=0;K<size;K++)
                        fscanf(fp,"%d",&map[I][K]);
}

int mark(int arr[30][30],int I,int J,int steps)
{
        if (I<0) return 0;
        if (I>=size) return 0;
        if (J<0) return 0;
        if (J>=size) return 0;

        if (map[I][J]==1 && arr[I][J]==-1) {
                arr[I][J]=steps;
                return 1;
        }
        return 0;
}

int meet(int I,int J)
{
        if (I<0) return 0;
        if (I>=size) return 0;
        if (J<0) return 0;
        if (J>=size) return 0;

        if (good[I][J]==bad[I][J] && bad[I][J]!=-1)
                return 1;
        return 0;
}

int origin(int arr[30][30],int I,int J,int steps)
{
        if (I<0) return 0;
        if (I>=size) return 0;
        if (J<0) return 0;
        if (J>=size) return 0;

        return (arr[I][J]==steps);
}

void play_back(int arr[30][30],int I,int J,int steps)
{
        if (steps==0)
                return;

        /* come from left */
        if (origin(arr,I,J-1,steps-1)) {
```



```
                play_back(arr,I,J-1,steps-1);
                fprintf(fout,"R");
        }
        /* come from right */
        else if (origin(arr,I,J+1,steps-1)) {
                play_back(arr,I,J+1,steps-1);
                fprintf(fout,"L");
        }
        /* come from up */
        else if (origin(arr,I-1,J,steps-1)) {
                play_back(arr,I-1,J,steps-1);
                fprintf(fout,"D");
        }
        /* come from down */
        else if (origin(arr,I+1,J,steps-1)) {
                play_back(arr,I+1,J,steps-1);
                fprintf(fout,"U");
        }
}

int play_good()
{
        int I,J,found;

        found=-1;

        for(I=0;I<size;I++) {
                for(J=0;J<size;J++) {
                   if (good[I][J]==good_steps) {
                        /* Up */
                        found+=mark(good,I-1,J,good_steps+1);
                        if (meet(I-1,J)) {
                                meet_I=I-1;
                                meet_J=J;
                                return good_steps+1;
                        }

                        found+=mark(good,I+1,J,good_steps+1);
                        if (meet(I+1,J)) {
                                meet_I=I+1;
                                meet_J=J;
                                return good_steps+1;
                        }

                        found+=mark(good,I,J-1,good_steps+1);
                        if (meet(I,J-1)) {
                                meet_I=I;
                                meet_J=J-1;
                                return good_steps+1;
                        }

                        found+=mark(good,I,J+1,good_steps+1);
                        if (meet(I,J+1)) {
                                meet_I=I;
                                meet_J=J+1;
                                return good_steps+1;
                        }
                   }
                }
        }

        good_steps++;
        return (found==-1)?-1:0;
}

int play_bad()
{
        int I,J,found;
```

```
        found=-1;

        for(I=0;I<size;I++) {
                for(J=0;J<size;J++) {
                   if (bad[I][J]==bad_steps) {
                        found+=mark(bad,I-1,J,bad_steps+1);
                        found+=mark(bad,I+1,J,bad_steps+1);
                        found+=mark(bad,I,J-1,bad_steps+1);
                        found+=mark(bad,I,J+1,bad_steps+1);
                   }
                }
        }

        bad_steps++;
        return found;
}

void init_bad()
{
        int I,J;

        for(I=0;I<size;I++)
                for(J=0;J<size;J++)
                        bad[I][J]=-1;
        bad[size-1][size-1]=0;
        bad_steps=0;
}


void init_good()
{
        int I,J;

        for(I=0;I<size;I++)
                for(J=0;J<size;J++)
                        good[I][J]=-1;
        good[0][0]=0;
        good_steps=0;
}

int main()
{
        int steps;

        fout=fopen("OUTPUT1.TXT","wt");
        read_input();
        init_bad();
        init_good();

        while(1) {
                if (play_bad()==-1) {
                        fprintf(fout,"NOPATH\n");
                        exit(0);
                }

                switch(steps=play_good()) {
                        case 0:
                                break;
                        case -1:
                                fprintf(fout,"NO PATH\n");
                                exit(0);
                        default:
                                fprintf(fout,"%d\n",steps);
                                play_back(good,meet_I,meet_J,steps);
                                fprintf(fout,"\n");
                                play_back(bad,meet_I,meet_J,steps);
                                fprintf(fout,"\n");
                                exit(0);
                }
```



```
        }
}
```