Sunteți rugați să ajutați la scrierea soft-ului pentru un robot autonom care navighează în interiorul unui labirint. Harta pe care robotul trebuie să se deplaseze este reprezentată sub forma unei matrice pătratice. În această matrice valorile de 0 reprezintă drum accesibil, iar 1 obstacole. Robotul nu poate să treacă peste obstacole.

Se citesc de la tastatură următoarele date, în ordine:

- un întreg n<=20, reprezentând dimensiunea unei matrice patratice ce va constitui harta labirintului în care robotul se va deplasa;
- n x n întregi, reprezentând valorile matricei - valorile de 1 reprezintă obstacole/margini, valorile de 0 reprezintă zonă navigabilă, iar valorile de 2 reprezintă bonus (valabil doar pentru subpunctul 4).
- o pereche de întregi reprezentând coordonatele punctului de plecare în labirint.
- un întreg, reprezentând o comandă, în funcție de valoarea căruia se vor executa diferite funcționalități, cum este menționat mai jos:

1. Se citesc suplimentar caractere individuale, până la întâlnirea EOF - aceste caractere reprezintă o secvență de instrucțiuni pe care o transmitem robotului și pot lua valorile: ‘u’ (up-sus), ‘r’ (right-dreapta), ‘d’ (down-jos), ‘l’ (left-stânga). Se afișează pe ecran instrucțiunile în ordinea citirii de la tastatură.
2. Se citesc suplimentar caractere individuale, până la întâlnirea EOF - aceste caractere reprezintă o secvență de instrucțiuni pe care o transmitem robotului și pot lua valorile: ‘u’ (up-sus), ‘r’ (right-dreapta), ‘d’ (down-jos), ‘l’ (left-stânga). Se afișează pe ecran coordonatele locului în care se oprește robotul după executarea tuturor instrucțiunilor. Dacă una dintre instrucțiuni îndreaptă robotul înspre o poziție nepermisă (obstacol), atunci execuția va sări peste acea instrucțiune și o va procesa pe următoarea.
3. Robotul se va îndrepta către ieșirea din labirint (o singură valoare de 0 pe marginea labirintului). Acesta navighează autonom, iar succesiunea de instrucțiuni pe care le va încerca la nivelul fiecărei celule este u->r->d->l. Este garantat că robotul nu va întâlni un drum închis sau bifurcații. Se afișează pe ecran coordonatele tuturor celulelor pe care robotul trebuie să le viziteze începând cu ieșirea pentru a se întoarce la punctul de plecare.
4. Pe harta apar anumite celule bonus care vor dubla amplitudinea mișcării executate atunci când celula este întâlnită, adică se vor parcurge 2 celule, în loc de una singură, dacă harta cu obstacole o permite. Altfel, se va executa un pas normal. Se afișează pe ecran coordonatele tuturor celulelor vizitate pana la ieșirea din labirint.

Exemplu:

1. Test #1

Input	Output
5 1 1 1 1 1 1 0 0 0 1 1 0 1 0 1 1 0 1 0 0 1 1 1 1 1 2 1 1 u r r u d d	u r r u d d
Explicație: s-au citit de la tastatură dimensiunea matricei (5), cele 25 de elemente ale matricei, indecșii celulei de plecare (2 1), comanda (1) și secvența de instrucțiuni. Se afișează pe ecran secvența de instrucțiuni în ordinea citirii de la tastatură.	

2. Test #2

Input	Output
5 1 1 1 1 1 1 0 0 0 1	3 3

1 0 1 0 1 1 0 1 0 0 1 1 1 1 1 2 1 2 u r r u d d	

Explicație: s-au citit de la tastatură dimensiunea matricei (5), cele 25 de elemente ale matricei, indecșii celulei de plecare (2 1), comanda (2) și secvența de instrucțiuni.
În continuare sunt evidențiate cu roșu pozițiile robotului.
Algoritmul începe din poziția de indecși [2][1].

1 1 1 1 1
1 0 0 0 1
1 0 1 0 1
1 0 1 0 0
1 1 1 1 1

Se execută instrucțiunea u:

1 1 1 1 1
1 0 0 0 1
1 0 1 0 1
1 0 1 0 0
1 1 1 1 1

Se execută instrucțiunea r:

1 1 1 1 1
1 0 0 0 1
1 0 1 0 1
1 0 1 0 0
1 1 1 1 1

Se execută instrucțiunea r:

1 1 1 1 1
1 0 0 0 1
1 0 1 0 1
1 0 1 0 0
1 1 1 1 1

Nu se poate execut instrucțiunea u, deoarece există un obstacol (marginea labirintului), deci robotul rămâne în aceeași poziție. Se trece la următoarea instrucțiune.

Se execută instrucțiunea d:

1 1 1 1 1
1 0 0 0 1
1 0 1 0 1
1 0 1 0 0
1 1 1 1 1

Se execută instrucțiunea d:

1 1 1 1 1
1 0 0 0 1
1 0 1 0 1
1 0 1 0 0
1 1 1 1 1

Se afișează pe ecran indecșii celulei în care a ajuns robotul, adică [3][3].

3. Test #3

Input	Output

5 1 1 1 1 1 1 0 0 0 1 1 0 1 0 1 1 0 1 0 0 1 1 1 1 1 2 1 3	3 4 3 3 2 3 1 3 1 2 1 1 2 1

Explicație: s-au citit de la tastatură dimensiunea matricei (5), cele 25 de elemente ale matricei, indecșii celulei de plecare (2 1) și comanda (3).
Algoritmul începe din poziția de indecși [2][1]. Secvența de instrucțiuni pe care o rulează algoritmul este u, r, r, d, d, r.

Pe drumul de întoarcere trebuie parcurse celulele în ordine inversă, afișând următorii indecși:
3 4
3 3
2 3
1 3
1 2
1 1
2 1

4. Test #4

Input	Output
5 1 1 1 1 1 1 2 0 0 1 1 0 1 2 1 1 0 1 0 0 1 1 1 1 1 2 1 3	2 1 1 1 1 3 2 3 3 3 3 4

Explicație: s-au citit de la tastatură dimensiunea matricei (5), cele 25 de elemente ale matricei, indecșii celulei de plecare (2 1) și comanda (4).
Algoritmul începe din poziția de indecși [2][1].

1 1 1 1 1
1 2 0 0 1
1 0 1 2 1
1 0 1 0 0
1 1 1 1 1

Se execută instrucțiunea u:

1 1 1 1 1
1 2 0 0 1
1 0 1 2 1
1 0 1 0 0
1 1 1 1 1

Se execută instrucțiunea r. Celula actuală are valoarea 2, deci reprezintă o celulă de bonus. Prin urmare, se va încerca deplasarea la dreapta cu 2 poziții:

1 1 1 1 1
1 2 0 0 1
1 0 1 2 1
1 0 1 0 0
1 1 1 1 1

Se execută instrucțiunea d:

1 1 1 1 1
1 2 0 0 1
1 0 1 2 1
1 0 1 0 0

```
1 1 1 1 1
```

Se execută instrucțiunea d. Celula actuală are valoarea 2, deci reprezintă o celulă de bonus. Prin urmare, se va încerca deplasarea în jos cu 2 poziții. Acest lucru nu este posibil, așa că se va face deplasarea normală, cu o singură poziție:

```
1 1 1 1 1
1 2 0 0 1
1 0 1 2 1
1 0 1 0 0
1 1 1 1 1
```

Se execută instrucțiunea r:

```
1 1 1 1 1
1 2 0 0 1
1 0 1 2 1
1 0 1 0 0
1 1 1 1 1
```

Se afișează pe ecran coordonatele tuturor celulelor vizitate până la ieșire:

```
2 1
1 1
1 3
2 3
3 3
3 4
```