Pele un siera klucis

3.izcilības (desmitnieka) uzdevums

Dots siera klucis, kas sagriezts pēc izmēra vienādos kubiciņos, bet ar dažādu masu. Siera kluča izmērus N x M x K kubiciņi un katra kubiciņa masu ievada no tastatūras. Pelei ir "jāizgraužas" cauri siera klucim no tā viena stūra (siera kubiciņa koordinātas (1,1,1)) līdz tā pretējam stūrim (siera kubiciņa koordinātas (n,m,k)). Pele vienā gājienā var apēst vienu siera kubiciņu, bet katrā nākamajā gājienā blakus esošo siera gabaliņu, kura masa ir ne mazāka kā iepriekšējā un atrodas koordinātu ass virzienā

Kods:

```
# Programmas nosaukums: Pele un siera klucis

# 3.izcilības (desmitnieka) uzdevums

# Uzdevuma formulējums: Dots siera klucis, kas sagriezts pēc izmēra vienādos kubiciņos, bet ar dažādu masu.

# Siera kluča izmērus N x M x K kubiciņi un katra kubiciņa masu ievada no tastatūras.

# Pelei ir "jāizgraužas" cauri siera klucim no tā viena stūra (siera kubiciņa koordinātas (1,1,1)) līdz tā pretējam stūrim (siera kubiciņa koordinātas (n,m,k)).

# Pele vienā gājienā var apēst vienu siera kubiciņu, bet katrā nākamajā gājienā blakus esošo siera gabaliņu,

# kura masa ir ne mazāka kā iepriekšējā un atrodas koordinātu ass virzienā.

# Programmas autors: Vladislavs Babaņins

# Versija 1.0



"""

Šī programma reprezentē 3D siera klucišu (paralēlskaldni), kurā pele mēģina nograuzt ceļu no viena siera stūra (1,1,1) uz pretējo stūri (n,m,k).

Siera kluciši sadala mazākos kubiciņos, katrs ar noteiktu masu.

Pele var grauzt no viena kuba uz blakus esošo tikai tad, ja blakus esošā kuba masa ir vienāda vai lielāka ar ieprēkšējo āpestu kubiciņu.

Lietotājs ievada siera bloka izmērus (N, M, K) un katra siera kubiciņa masu.

Pēc tam programma atrod ceļu no sākuma stūra (1,1,1) uz pretējo stūri, ja tāds ceļš vispār eksistē (vienkārši parāda pirmo atrāstu ceļu).
```

Ceļš tiek parādīts kā peles veikto kustību secība.

Ja nav derīga ceļa, programma norāda, ka pele nevarēja "legāli" izgrauzt siera klucišu.

"""

```
import numpy as np

import copy



def eat_cheese(block, x, y, z, path, current_mass):

    # Peles ēšanas funkcija.

    # Rekursīva funkcija, lai noteiktu ceļu, pa kuru pele var iet cauri siera bloka, no sākumpunkta (1,1,1) līdz (N,M,K), tā lai tas būtu "legāli" (no māzaka uz lielāka kubiciņa).

    # Funkcija pēta visus iespējamos ceļus izmantojot tā saucāmo "backtracking" algoritmu.

    # Mēs vienmēr ejam vienā virzienā (piemēram x), pec tam, ja tur ir strūpceļš (ir tikai mazākie siera kubiciņi), tad atkāpjamies pa vienu soli, un jau ejam citā virziena.

    # Tāda veidā arī mēs realizējam kad ir bijis 2D labirints, šeit ir tas pats bet 3D.

    # Ja pele sasniedz siera bloka pretējo stūri (N,M,K), tiek atgriezts pašreizējais ceļš.

    # Ja pele izmēģinājusi visus gadījumus un nav nevienas "legālas" izejas, tad tiek atgriezts None.

    # block - trīsdimensijas masīvs, kas attēlo siera kluciti. Katrs masīva elements ir kubiciņš, kas attēlo attiecīgā siera kuba masu.

    # x, y, z - pašreizējās peles koordinātas siera blokā.

    # x no 0 līdz N-1, y no 0 līdz M-1 un z no 0 līdz K-1 (Kur N, M, K ir siera bloka izmēri).

    # Šīs koordinātas tiek izmantotas, lai indeksētu kubiciņus masīvā, lai iegūtu pašreizējā siera kubiciņa masu.

    # x - x koordināta

    # y - y koordināta

    # z - z koordināta

    # path - kortežu saraksts, kur katrs kortežs apzīmē siera kuba koordinātas, ko pele ir izgrauzusi.

    # Korteži ir tādā secībā, kādā pele apēda kubiciņus.

    # Piemēram, ceļš [(0,0,0), (0,1,0), (1,1,0)] nozīmē, ka pele sākās no kuba (0,0,0), pēc tam tika pārvietota uz kubu ar koordinātiem (0,1,0) un beidzot pārvietots uz kubu ar koordinātiem (1,1,0).
```

```
    # Ceļš vienmēr sākas ar (0,0,0), jo pele vienmēr sāka no šī stura.
    # current_mass - pašreizējā siera kubiciņa masa. Pele var pārvietoties tikai uz siera kubu, kura masa ir vienāda vai lielāka par pašreizējo kubiciņa masu.
    # Tas nodrošina, ka pele vienmēr pāriet uz siera kubu ar vienādu vai lielāku masu.

    # Ja esam sasnieguši galamērķi (N,M,K), atgriezam ceļu un beidzās funkcija.
    if x == N - 1 and y == M - 1 and z == K - 1:
        return path

    # Ja mēs varam virzīties x virzienā... (āpest kubu kas ir ar X koordinātiem).
    if x + 1 < N and block[x + 1][y][z] >= current_mass:  # Varam virzities tikai, ja ir lielāka vai vienāda masa nekā pašreizeja kubiciņa masa.
        path_copy = copy.deepcopy(path)  # Izveidojam līdzšinējā ceļa kopiju (jo citādi varētu izmainīties arī nenokopēts ceļš).
        path_copy.append((x + 1, y, z))  # Pievienojam jaunu pozīciju kopētajam ceļam.
        # Rekursīvi sākam est siera kubiciņus no jaunās pozīcijas.
        result = eat_cheese(block, x + 1, y, z, path_copy, block[x + 1][y][z])
        # Ja ir atrasts derīgs ceļš, atgriezisim to.
        if result is not None:
            return result

    # Ja mēs varam virzīties y virzienā... (jau x virzienu esam izmeģinājusi).
    if y + 1 < M and block[x][y + 1][z] >= current_mass:  # Varam virzities tikai, ja ir lielāka vai vienāda masa nekā pašreizeja kubiciņa masa.
        path_copy = copy.deepcopy(path)  # Izveidojam līdzšinējā ceļa kopiju (jo citādi varētu izmainīties arī nenokopēts ceļš).
        path_copy.append((x, y + 1, z))  # Pievienojam jaunu pozīciju kopētajam ceļam.
        # Rekursīvi sākam est siera kubiciņus no jaunās pozīcijas.
        result = eat_cheese(block, x, y + 1, z, path_copy, block[x][y + 1][z])
        if result is not None:
            return result

    # Ja mēs varam virzīties z virzienā... (jau x un y virzienu esam izmeģinājusi).
```

```
    if z + 1 < K and block[x][y][z + 1] >= current_mass:  # Varam virzities tikai, ja ir lielāka vai vienāda masa nekā pašreizeja kubiciņa masa.
        path_copy = copy.deepcopy(path)  # Izveidojam līdzšinējā ceļa kopiju (jo citādi varētu izmainīties arī nenokopēts ceļš).
        path_copy.append((x, y, z + 1))  # Pievienojam jaunu pozīciju kopētajam ceļam.
        # Rekursīvi sākam est siera kubiciņus no jaunās pozīcijas.
        result = eat_cheese(block, x, y, z + 1, path_copy, block[x][y][z + 1])
        if result is not None:
            return result

    # Ja nebija neviena derīga ceļa, atgriežam None.
    return None


# ------------------------------------------------------------
# Galvenā programmas daļa
# ------------------------------------------------------------


# Prasam lietotājam ievadīt siera kluciša izmērus (naturāli skaitļi).
N = int(input("Ievadiet siera kluciša garumu N ==> "))  # Cik kubiciņu garš ir klucitis (cik ir N kubiciņu).
M = int(input("Ievadiet siera kluciša platumu M ==> "))  # Cik kubiciņu plats ir klucitis (cik ir M kubiciņu).
K = int(input("Ievadiet siera kluciša augstumu K ==> "))  # Cik kubiciņu augsts ir klucitis (cik ir K kubiciņu).
print()

# Izveidojam "tukšu" siera kluciti.
cheese_block = np.zeros((N, M, K))  # Izveidojam trīsdimensijas masīvu, kas piepildīts ar nullēm.

# Prasam lietotājam ievadīt katra kubiciņa masu.
```

```
for i in range(N):  # Katrai x-koordinātai ...
    for j in range(M):  # ... un katrai y-koordinātai ...
        for k in range(K):  # ... un katrai z-koordinātai ...
            # ... prasam lietotājam ievadīt kubiciņu masu šajā konkrēta vietā (ar konkrētam koordinātem)
            cheese_block[i][j][k] = float(input(f"Ievadiet kubiciņa ({i+1} x {j+1} x {k+1}) masu ==> "))

path = eat_cheese(cheese_block, 0, 0, 0, [(0, 0, 0)], cheese_block[0][0][0])  # Sākam no koordinātas (0,0,0).

if path is not None:  # Izvadam ceļu, ja tas nav None (ja eksistē ceļš).
    print("\nPele apēdusi siera kubiciņus un sasniedza pretējo galu izmantojot šādu ceļu:")
    move = 1
    for cube in path:
        print(f"{move}.gājiens: Apēst siera klucīšu ar koordinātām {cube[0] + 1, cube[1] + 1, cube[2] + 1}")  # x, y, z koordinātas.
        move = move + 1
else:
    print("\nPele nevar sasniegt siera kluciša pretējo galu!")

'''
TESTA PIEMĒRI
# --------------------------

N = 3, M = 3, K = 3
Siera kubiciņu masas: 1, 2, 3, 2, 3, 4, 3, 4, 5, 4, 5, 6, 5, 6, 7, 6, 7, 8, 7, 8, 9, 8, 9, 10, 9, 10, 11
Izvada:
Pele apēdusi siera kubiciņus un sasniedza pretējo galu izmantojot šādu ceļu:
1.gājiens: Apēst siera klucīšu ar koordinātām (1, 1, 1)
2.gājiens: Apēst siera klucīšu ar koordinātām (2, 1, 1)
3.gājiens: Apēst siera klucīšu ar koordinātām (3, 1, 1)
```

4.gājiens: Apēst siera klucīšu ar koordinātām (3, 2, 1)

5.gājiens: Apēst siera klucīšu ar koordinātām (3, 3, 1)

6.gājiens: Apēst siera klucīšu ar koordinātām (3, 3, 2)

7.gājiens: Apēst siera klucīšu ar koordinātām (3, 3, 3)

N = 2, M = 2, K = 2

Siera kubiciņu masas: 1, 1, 1, 1, 1, 1, 1, 1

Pele apēdusi siera kubiciņus un sasniedza pretējo galu izmantojot šādu ceļu:

1.gājiens: Apēst siera klucīšu ar koordinātām (1, 1, 1)

2.gājiens: Apēst siera klucīšu ar koordinātām (2, 1, 1)

3.gājiens: Apēst siera klucīšu ar koordinātām (2, 2, 1)

4.gājiens: Apēst siera klucīšu ar koordinātām (2, 2, 2)

N = 1, M = 1, K = 1

Siera kubiciņu masas: 1

Izvada:

Pele apēdusi siera kubiciņus un sasniedza pretējo galu izmantojot šādu ceļu:

1.gājiens: Apēst siera klucīšu ar koordinātām (1, 1, 1)

N = 2, M = 2, K = 2

Siera kubiciņu masas: 1, 2, 2, 3, 2, 3, 3, 4

Pele apēdusi siera kubiciņus un sasniedza pretējo galu izmantojot šādu ceļu:

1.gājiens: Apēst siera klucīšu ar koordinātām (1, 1, 1)

2.gājiens: Apēst siera klucīšu ar koordinātām (2, 1, 1)

3.gājiens: Apēst siera klucīšu ar koordinātām (2, 2, 1)

4.gājiens: Apēst siera klucīšu ar koordinātām (2, 2, 2)

N = 2, M = 2, K = 2

Siera kubiciņu masas: 1, 2, 2, 1, 2, 1, 1, 2

Pele nevar sasniegt siera kluciša pretējo galu!

'''

Testa piemēri:

1)

```
Ievadiet siera kluciša garumu N ==> 2
Ievadiet siera kluciša platumu M ==> 2
Ievadiet siera kluciša augstumu K ==> 2

Ievadiet kubiciņa (1 x 1 x 1) masu ==> 1
Ievadiet kubiciņa (1 x 1 x 2) masu ==> 1
Ievadiet kubiciņa (1 x 2 x 1) masu ==> 1
Ievadiet kubiciņa (1 x 2 x 2) masu ==> 1
Ievadiet kubiciņa (2 x 1 x 1) masu ==> 1
Ievadiet kubiciņa (2 x 1 x 2) masu ==> 1
Ievadiet kubiciņa (2 x 2 x 1) masu ==> 1
Ievadiet kubiciņa (2 x 2 x 2) masu ==> 1

Pele apēdusi siera kubiciņus un sasniedza pretējo galu izmantojot šādu ceļu:
1.gājiens: Apēst siera klucīšu ar koordinātām (1, 1, 1)
2.gājiens: Apēst siera klucīšu ar koordinātām (2, 1, 1)
3.gājiens: Apēst siera klucīšu ar koordinātām (2, 2, 1)
4.gājiens: Apēst siera klucīšu ar koordinātām (2, 2, 2)
```

2)

```
Ievadiet siera kluciša garumu N ==> 2
Ievadiet siera kluciša platumu M ==> 2
Ievadiet siera kluciša augstumu K ==> 2

Ievadiet kubiciņa (1 x 1 x 1) masu ==> 1
Ievadiet kubiciņa (1 x 1 x 2) masu ==> 2
Ievadiet kubiciņa (1 x 2 x 1) masu ==> 2
Ievadiet kubiciņa (1 x 2 x 2) masu ==> 1
Ievadiet kubiciņa (2 x 1 x 1) masu ==> 2
Ievadiet kubiciņa (2 x 1 x 2) masu ==> 1
Ievadiet kubiciņa (2 x 2 x 1) masu ==> 1
Ievadiet kubiciņa (2 x 2 x 2) masu ==> 2

Pele nevar sasniegt siera kluciša pretējo galu!
```

3)

```
Ievadiet siera klucīša garumu N ==> 3
Ievadiet siera klucīša platumu M ==> 3
Ievadiet siera klucīša augstumu K ==> 3

Ievadiet kubiciņa (1 x 1 x 1) masu ==> 1
Ievadiet kubiciņa (1 x 1 x 2) masu ==> 2
Ievadiet kubiciņa (1 x 1 x 3) masu ==> 3
Ievadiet kubiciņa (1 x 2 x 1) masu ==> 2
Ievadiet kubiciņa (1 x 2 x 2) masu ==> 3
Ievadiet kubiciņa (1 x 2 x 3) masu ==> 4
Ievadiet kubiciņa (1 x 3 x 1) masu ==> 3
Ievadiet kubiciņa (1 x 3 x 2) masu ==> 4
Ievadiet kubiciņa (1 x 3 x 3) masu ==> 5
Ievadiet kubiciņa (2 x 1 x 1) masu ==> 4
Ievadiet kubiciņa (2 x 1 x 2) masu ==> 5
Ievadiet kubiciņa (2 x 1 x 3) masu ==> 6
Ievadiet kubiciņa (2 x 2 x 1) masu ==> 5
Ievadiet kubiciņa (2 x 2 x 2) masu ==> 6
Ievadiet kubiciņa (2 x 2 x 3) masu ==> 7
Ievadiet kubiciņa (2 x 3 x 1) masu ==> 6
Ievadiet kubiciņa (2 x 3 x 2) masu ==> 7
Ievadiet kubiciņa (2 x 3 x 3) masu ==> 8
Ievadiet kubiciņa (3 x 1 x 1) masu ==> 7
Ievadiet kubiciņa (3 x 1 x 2) masu ==> 8
Ievadiet kubiciņa (3 x 1 x 3) masu ==> 9
Ievadiet kubiciņa (3 x 2 x 1) masu ==> 8
Ievadiet kubiciņa (3 x 2 x 2) masu ==> 9
Ievadiet kubiciņa (3 x 2 x 3) masu ==> 10
Ievadiet kubiciņa (3 x 3 x 1) masu ==> 9
Ievadiet kubiciņa (3 x 3 x 2) masu ==> 10
Ievadiet kubiciņa (3 x 3 x 3) masu ==> 11

Pele apēdusi siera kubiciņus un sasniedza pretējo galu izmantojot šādu ceļu:
1.gājiens: Apēst siera klucīšu ar koordinātām (1, 1, 1)
2.gājiens: Apēst siera klucīšu ar koordinātām (2, 1, 1)
3.gājiens: Apēst siera klucīšu ar koordinātām (3, 1, 1)
4.gājiens: Apēst siera klucīšu ar koordinātām (3, 2, 1)
5.gājiens: Apēst siera klucīšu ar koordinātām (3, 3, 1)
6.gājiens: Apēst siera klucīšu ar koordinātām (3, 3, 2)
7.gājiens: Apēst siera klucīšu ar koordinātām (3, 3, 3)
```

4)

```
Ievadiet siera kluciša garumu N ==> 1
Ievadiet siera kluciša platumu M ==> 1
Ievadiet siera kluciša augstumu K ==> 1

Ievadiet kubiciņa (1 x 1 x 1) masu ==> 1

Pele apēdusi siera kubiciņus un sasniedza pretējo galu izmantojot šādu ceļu:
1.gājiens: Apēst siera klucīšu ar koordinātām (1, 1, 1)
```

5)

```
Ievadiet siera kluciša garumu N ==> 2
Ievadiet siera kluciša platumu M ==> 2
Ievadiet siera kluciša augstumu K ==> 2

Ievadiet kubiciņa (1 x 1 x 1) masu ==> 1
Ievadiet kubiciņa (1 x 1 x 2) masu ==> 2
Ievadiet kubiciņa (1 x 2 x 1) masu ==> 2
Ievadiet kubiciņa (1 x 2 x 2) masu ==> 3
Ievadiet kubiciņa (2 x 1 x 1) masu ==> 2
Ievadiet kubiciņa (2 x 1 x 2) masu ==> 3
Ievadiet kubiciņa (2 x 2 x 1) masu ==> 3
Ievadiet kubiciņa (2 x 2 x 2) masu ==> 4

Pele apēdusi siera kubiciņus un sasniedza pretējo galu izmantojot šādu ceļu:
1.gājiens: Apēst siera klucīšu ar koordinātām (1, 1, 1)
2.gājiens: Apēst siera klucīšu ar koordinātām (2, 1, 1)
3.gājiens: Apēst siera klucīšu ar koordinātām (2, 2, 1)
4.gājiens: Apēst siera klucīšu ar koordinātām (2, 2, 2)
```