44. Bundeswettbewerb Informatik - 1. Runde / 2. Aufgabe

Luna Hagemann (Team-ID 00008 (Einzelteam), Teilnahme-ID 78245)

17. November 2025

Inhaltsverzeichnis

1 Lösungsidee

Dadurch, dass die Daten noch nicht zu groß sind, funktioniert ein Brute-Force-Algorithmus, um die entsprechenden Choreos zu finden. Dieser Algorithmus nimmt eine Taktzahl als Argument und gibt eine Choreographie ab diesem Zeitpunkt zurück. Jede mögl. Figur wird jetzt in diesem Level durchgegangen und jeweils von der insgesamten Taktzahl abgezogen. Wäre die Taktzahl dann negativ, wird die Figur übersprungen und die nächste wird überprüft. Ist die Taktzahl positiv, wird mit den verbleibenden Takten wieder die Funktion aufgerufen und die zurückgegebenen Choreographien werden dann jeweils mit der aktuellen Figur als Choreographie gespeichert. Ist die Taktzahl genau 0, wird überprüft, ob jede*r Tänzer*in wieder am eigenen Platz ist und, wenn dies der Fall ist, diese Figur als Choreographie gespeichert. Nachdem alle Figuren so geprüft wurden, werden alle gespeicherten Choreographien zurückgegeben. So kommt jede mögl. Choreographie bis an den Anfang. Sind alle erlaubten Choreographien erstellt, wird geguckt, welche nach den Kriterien am besten sind, bzw. es wird ausgegeben, dass keine Choreos gefunden werden konnten.

2 Umsetzung

Nachdem der Input verarbeitet wird, werden alle Figuren effizient in einer globalen Liste gespeichert. Damit die rekursive Funktion gut läuft, muss neben der Taktzahl auch noch die Liste aller Figuren als Argument übermittelt werden, sowie die aktuelle Position, diesmal als Liste von Zahlen von 0 bis 15 (0 stand am Anfang ganz links, 15 ganz rechts), diese wird dann immer mit der aktuellen Figur verändert, um am Ende zu prüfen, ob die Choreographie auch am Ende gleich endet. Figuren werden generell als dictonary gespeichert, mit den Feldern 'name', 'bars' (Anzahl an Takten (engl. bars)) und 'pos_alph' (Position in alphabetischer Schreibweise).

3 Beispiele

3.1 choreo01

	Anzahl	Figuren
max. versch. Figuren	2	Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Up, Cast_Up, Cast_Up, Cast_Up
max. Figuren	16	Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four
min. Figuren	8	Cast_Up, Cast_Up, Cast_Up, Cast_Up, Cast_Up, Cast_Up, Cast_Up, Cast_Up
max. Distanz	384	Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four, Cast_Four
min. Distanz	240	Cast_Up, Cast_Up, Cast_Up, Cast_Up, Cast_Up, Cast_Up, Cast_Up, Cast_Up

3.2 choreo02

	Anzahl	Figuren
max. versch. Figuren	4	Cycle_1, Cycle_1, Cycle_1, Cycle_2, Cycle_2, Cycle_2, Cycle_3, Cycle_4, Cycle_4, Cycle_4
max. Figuren	19	Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_2, Cycle_2, Cycle_4
min. Figuren	8	Cycle_3, Cycle_3, Cycle_3, Cycle_3, Cycle_3, Cycle_3, Cycle_3, Cycle_3
max. Distanz	688	Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_2, Cycle_2, Cycle_4
min. Distanz	624	Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_1, Cycle_4, Cycle_4, Cycle_4, Cycle_4

3.3 choreo03

Keine legale Choreographie konnte gefunden werden.

3.4 choreo04

Keine legale Choreographie konnte gefunden werden.

3.5 choreo05

	Anzahl	Figuren
max. versch. Figuren	1	Hip_Twist, Hip_Twist
max. Figuren	24	Hip_Twist, Hip_Twist
min. Figuren	24	Hip_Twist, Hip_Twist
max. Distanz	1728	Hip_Twist, Hip_Twist
min. Distanz	1728	Hip_Twist, Hip_Twist

3.6 choreo06

	Anzahl	Figuren
max. versch. Figuren	4	Cycle_4, Cycle_4, Cycle_4, Pairs, Cycle_Back, Cycle_Back, Pairs, Cycle_Back, Sides, Cycle_Back
max. Figuren	16	Pairs, Pairs, Pairs, Pairs, Pairs, Pairs, Pairs, Cycle_Back, Cycle_Back, Cycle_Back, Cycle_Back, Cycle_Back, Cycle_Back, Cycle_Back, Cycle_Back, Pairs
min. Figuren	4	Sides, Sides, Sides, Sides
max. Distanz	768	Cycle_4, Cycle_4, Cycle_4, Cycle_4, Cycle_4, Cycle_4, Cycle_4, Cycle_4
min. Distanz	368	Pairs, Pairs, Pairs, Pairs, Pairs, Pairs, Pairs, Cycle_Back, Cycle_Back, Cycle_Back, Cycle_Back, Cycle_Back, Cycle_Back, Cycle_Back, Cycle_Back, Pairs

3.7 choreoA

Keine legale Choreographie konnte gefunden werden.

4 Quelltext

```
alphabet = {
    'A': 0,
    'B': 1,
    'C': 2,
    'D': 3,
    'E': 4,
    'F': 5,
    'G': 6,
    'H': 7,
    'I': 8,
    'J': 9,
    'K': 10,
    'L': 11,
    'M': 12,
    'N': 13,
    'O': 14,
    'P': 15,
}
gpos = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]


def explore(bars: int, figs: list[dict[str, str | int]], pos: list[int]) -> list[tuple[dict[str, str | int]]]:
    choreos = []
    for fig in figs:
        rem = bars
        rem -= fig['bars']
        if rem == 0:
            if apply_fig(pos, fig['pos_alph']) == gpos:
                choreos.append((fig,))
        elif rem < 0:
            continue
        else:
            pos = apply_fig(pos, fig['pos_alph'])
            ret = explore(rem, figs, pos)
            for choreo in ret:
                choreos.append((fig, *choreo))
    return choreos


def apply_fig(pos: list[int], fig: str) -> list[int]:
    new_pos = []
    for char in fig:
        new_pos.append(pos[alphabet[char]])
    return new_pos

def calc_distance(fig: str) -> int:
    distance = 0
    for i, char in enumerate(fig):
        distance += abs(alphabet[char] - i)
    return distance


with open('./inp/choreo06.txt', 'r') as f:
    bars = int(f.readline().strip())
    figs = []
    for _ in range(int(f.readline().strip())):
        raw = f.readline().strip().split('-')
```

```
        figs.append({'name': raw[0], 'bars': int(raw[1]), 'pos_alph': raw[2]})

choreos = explore(bars, figs, gpos)

if not choreos:
    print('No valid choreo found.')
    exit()

high = {
    'max_dif': [0, None],
    'max': [0, None],
    'min': [float('inf'), None],
    'max_dis': [0, None],
    'min_dis': [float('inf'), None],
}

for choreo in choreos:
    figs = []
    n_dif = 0
    n_figs = 0
    dis = 0
    for fig in choreo:
        dis += calc_distance(fig['pos_alph'])
        n_figs += 1
        if not fig['name'] in figs:
            figs.append(fig['name'])
            n_dif += 1
    if n_dif > high['max_dif'][0]:
        high['max_dif'] = [n_dif, choreo]
    if n_figs > high['max'][0]:
        high['max'] = [n_figs, choreo]
    if n_figs < high['min'][0]:
        high['min'] = [n_figs, choreo]
    if dis > high['max_dis'][0]:
        high['max_dis'] = [dis, choreo]
    if dis < high['min_dis'][0]:
        high['min_dis'] = [dis, choreo]

# OUTPUT
```