sequencingproblem

```
# -*- coding: utf-8 -*-
"""sequencingproblem.ipynb

Automatically generated by Colaboratory.

Original file is located at
https://colab.research.google.com/drive/1pp1u6ceRVlFBt9gKeyxF1Pht7958zZk4
"""

import itertools

def calculate_elapsed_time(sequence, jobs):
machine_1_time = 0
machine_2_time = 0

for job_idx in sequence:
job = jobs[job_idx]
machine_1_time += job[0]
machine_2_time = max(machine_1_time, machine_2_time) + job[1]

return machine_2_time


def find_optimal_sequence(jobs):
job_indices = list(range(len(jobs)))

# Generate all possible permutations of the jobs
permutations = list(itertools.permutations(job_indices))

optimal_sequence = None
min_elapsed_time = float('inf')

for perm in permutations:
elapsed_time = calculate_elapsed_time(perm, jobs)

if elapsed_time < min_elapsed_time:
min_elapsed_time = elapsed_time
optimal_sequence = perm
```

```
return optimal_sequence, min_elapsed_time


# User input for number of jobs
num_jobs = int(input("Enter the number of jobs: "))

jobs = []
for i in range(num_jobs):
job_name = input("Enter job name: ")
machine_a_duration = int(input("Enter duration for Machine A: "))
machine_b_duration = int(input("Enter duration for Machine B: "))
jobs.append((machine_a_duration, machine_b_duration, job_name))

optimal_sequence, total_elapsed_time = find_optimal_sequence(jobs)

print("Optimal Sequence:", optimal_sequence)
print("Total Elapsed Time:", total_elapsed_time)
```