# Junioraufgabe 2: Treffsicherheit

Team-ID: 00450 Team: Die Lamas

Bearbeiter/-innen dieser Aufgabe:

Philipp Tiede

1.November 2021

## Inhaltsverzeichnis



## Lösungsidee

Meine Idee wie ich dieses Problem lösen kann war, dass ich erst für jede Person ausrechne, wieviel Termine für jede Präferenz geändert werden müssen (sowohl den Termin dieser Präferenz inbegriffen (-> 2) und nicht inbegriffen (-> 1)), damit dieser der beste Termin ist.

Dann gucke ich für jeden Termin, was die Summe der verschieden Terminänderungen der Personen sind. Der Termin mit den wenigsten Terminänderungen ist dann das Ergebnis.

## Umsetzung 1 (nicht inbegriffen)

Um dies zu erreichen, speichere ich erst die Termine mit Präferenzen in ein 2-dimensionales Array. Dann ersetzt ich jede Präferenz mit der Kost (der benötigten Menge an Terminveränderungen). Die Menge an Terminveränderungen ist auch die Position (ab 0) vom ersten Vorkommen des Präferenzwertes in einer sortierten Tabelle der Termine dieser Person.

Als nächsten Schritt sortiere ich das 2D-Array nach Termin und nichtmehr Person, damit ich alle benötigten Terminänderungen pro Termin zusammenaddieren kann.

Zuletzt gebe ich den Wert und die Terminnummer von dem Termin mit dem niedrigsten Wert an den Nutzer zurück.

## Umsetzung 2 (inbegriffen)

Da ich die Umsetzung 1 zuerst geschrieben habe basiert die 2. Lösung auf dieser. Um die Termine so zu ändern, das auch der Termin, der Präferenz inbegriffen ist, ändere ich einfach die Menge an Terminänderungen bei allen, wo die normale Menge an Terminänderungen >= 1 ist auf 1, da ich jede Anzahl an Terminänderungen auf ein ändern kann, indem ich den Termin der Präferenz auf Grün ändere.

## Beispiele 1

```
> python main1.py praeferenzen0.txt
Für den 6. Termin müssen 2 Einträge verändert werden.
```

---

```
> python main1.py praeferenzen1.txt
Für den 2. Termin müssen 3 Einträge verändert werden.
```

```
> python main1.py praeferenzen2.txt
Für den 4. Termin müssen 0 Einträge verändert werden.
```

```
> python main1.py praeferenzen3.txt
Für den 11. Termin müssen 41 Einträge verändert werden.
```

```
> python main1.py praeferenzen4.txt
Für den 2. Termin müssen 157 Einträge verändert werden.
```

```
> python main1.py praeferenzen5.txt
Für den 63. Termin müssen 930 Einträge verändert werden.
```

# Beispiele 2

```
> python main2.py praeferenzen0.txt
Für den 6. Termin müssen 2 Einträge verändert werden.
```

```
> python main2.py praeferenzen1.txt
Für den 2. Termin muss 1 Eintrag verändert werden.
```

```
> python main2.py praeferenzen2.txt
Für den 4. Termin müssen 0 Einträge verändert werden.
```

```
> python main2.py praeferenzen3.txt
Für den 18. Termin müssen 7 Einträge verändert werden.
```

```
> python main2.py praeferenzen4.txt
Für den 22. Termin müssen 14 Einträge verändert werden.
```

```
> python main2.py praeferenzen5.txt
Für den 31. Termin müssen 34 Einträge verändert werden.
```

# Quellcode

```
def neededChange(dates:List[int]):
  """
    Für die Liste an Verfügbarkeiten die benötigten Änderungen
    für jedes Datum ausrechnen
  """
  neededChanges = []
  for i in dates:
    # Für jedes Datum sind die benötigten Änderungen
    # die Anzahl an Werten vor dem niedrigsten vorkommen
    # dieses Wertes in einem sortierten Array
    neededChanges.append(sorted(dates).index(i))
  return neededChanges

def main() -> None:
  """
    main
  """
  dataFileLocation = sys.argv[1]        #
  dataFile = open(dataFileLocation, "r")  # Datei einlesen und in data als String
  data = dataFile.read()                # speichern.


  # Aus Datei eine 2D Array erstellen
  dates = []
  for i in range(int(data.split(" ")[0])):
    preferences = data.splitlines()[i + 1]
    preference = []
    for j in range(int(data.split(" ")[1].split("\n")[0])):
      preference.append(int(preferences.split(" ")[j]))
```

```
  dates.append(preference)


# Für jede Person die benötigten Änderungen speichern
dates = [neededChange(i) for i in dates]


# Änderungen sind maximal 1, da der Termin immer selbst geändert werden kann
dates = [[0 if j == 0 else 1 for j in i] for i in dates] # Lösung 2


# Nach Termin anstatt Person sortieren.
dates = list(zip(*dates))

# Die Summe der benötigten Terminänderungen speichern
dates = [sum(i) for i in dates]

# Für den Termin mit der kleinsten Menge an Terminänderungen
# die Anzahl an Terminänderungen und die Terminnummer spiechern
solutionCost = min(dates)
solutionDate = dates.index(solutionCost)

# Als formatierten String ausgeben
print("Für den {solutionDate}. Termin {muss} {solutionCost} {eintrag} verändert werden."
  .format(solutionDate=solutionDate+1, solutionCost=solutionCost,
  # Gramatik :)
  muss="muss" if solutionCost == 1 else "müssen",
  eintrag="Eintrag" if solutionCost == 1 else "Einträge"))
```