



```
In [1]: import pandas as pd
import numpy as np
```

```
In [4]: df = pd.read_excel("fountains.xlsx")
df.head()
```

```
Out[4]:
```

	x	y	location
0	2.397371	1.889389	Левобережный
1	2.518151	3.218424	Левобережный
2	1.812677	1.812690	Левобережный
3	3.263370	2.613948	Левобережный
4	1.624420	2.434048	Левобережный

```
In [5]: def dist(x, y, fontains):
        dists = []
        for X, Y in fontains:
            d = np.sqrt((x - X)**2 + (y - Y)**2)
            dists.append(d)

        return min(dists)
```

```
In [6]: schems = {
        1: [(5, 3.5)],
        2: [(2, 2), (8, 2)],
        3: [(2, 2), (8, 2), (5, 5)]
        }

for s, coords in schems.items():
    df[f"d_{s}"] = df.apply(lambda row: dist(row.x, row.y, coords), axis = 1)
    df[f"U_{s}"] = 10 / (2 + df[f"d_{s}"])

for s in schems:
    print(f"Схема {s}: {df[f'U_{s}'].mean():.4f}")
```

Схема 1: 2.0645
Схема 2: 3.0899
Схема 3: 3.5033

Ответ: схема номер 3 максимизирует среднюю полезность

```
In [7]: def getvote(row):
        U1, U2, U3 = row["U_1"], row["U_2"], row["U_3"]
        B_u = max(U1, U2, U3)
        if (U1 == B_u): return 1
        if (U2 == B_u): return 2
        return 3
```

```
In [9]: df["vote"] = df.apply(getvote, axis=1)
```

```
print(df["vote"].value_counts())
```

```
vote
2    392
3     93
1     15
Name: count, dtype: int64
```

Ответ: схема 2 должна выиграть за нее голосует абсолютное большинство.

```
In [10]: A_def = 20
cost_fountain = 1500

for s in schems:
    num_fountain = len(schems[s])
    df[f"vtp_s"] = A_def / (2 + df[f"d_{s}"])
    total = df[f"vtp_s"].sum()
    profit = total - num_fountain * cost_fountain
    print(f"Схема {s}: {profit:.2f}")
```

```
Схема 1: 564.46
Схема 2: 89.93
Схема 3: -996.65
```

Компания выберет схему 1 так как именно она приносит наибольшую прибыль.

```
In [18]: like_coords = [(5, 5.5), (4.5, 4.5), (5.5, 4.5)]
df["d_l"] = df.apply(lambda row: dist(row.x, row.y, like_coords), axis = 1)
others = df[df["location"] != "Нагорье"]
nagorie = df[df["location"] == "Нагорье"]

rev_others = (A_def / (2 + others["d_3"])).sum()
rev_othersnag = (A_def / (2 + others["d_l"])).sum()

coeff_base = (1 / (2 + nagorie["d_3"])).sum()
coeff_nag = (1 / (2 + nagorie["d_l"])).sum()

delta_coeff = coeff_nag - coeff_base
delta_val = rev_others - rev_othersnag

print(f"{delta_coeff:.4f}")
print(f"{delta_val:.4f}")
print(delta_val / delta_coeff)
```

```
3.1728
1298.8605
409.37433810683723
```

Ответ: жители Нагорья должны выбрать значения A_i больше 409.37433810683723