In [ ]:
```
#vraag 1
#geen antwoord
```

In [ ]:
```
#import tips van seaborn
import seaborn as sns

tips = sns.load_dataset("tips")

print(tips.head())

## geen van bijde werkt voor import bij mij

from seaborn import datasets

tips = datasets.load_tips()

print(tips.head())
```

In [ ]:
```
#vraag 2
import matplotlib.pyplot as plt
import pandas as pd
bar_data = tips.days.value_counts().head(10)
bar_plot = bar_data.plot.bar(figsize=(5,5), title = 'dagen dat de mensen het m
```

In [ ]:
```
#vraag 3
labels = 'Rookers' , 'Niet Rokers'
plt.pie(tips.smoker.value_counts(), labels = labels, autopct='%3.1f%%')
plt.title('Cirkeldiagram van smokers')
plt.show()
```

In [18]:
```
#vraag 4
pd.crosstab(tips.day, tips.tip).plot(kind = 'bar', rot = 0)
```

In [19]:
```
#vraag 5
pd.crosstab(tips.smoker, tips.sex).plot(kind = 'bar', rot = 0)
```

In [20]:
```
#vraag 6
pd.crosstab(tips.tip, tips.time, margins = True)
#Bespreek
```

In [21]:
```
#vraag 7
pd.crosstab(tips.tip, tips.total_bill, margins = True)
#Bespreek
```

In [22]:
```
#vraag 8
data = tips.head(5)
data.notnull().boxplot('Feature total_bill')
```

In [23]:
```
#vraag 9
pd.crosstab(tips.sex, tips.day, margins = True)
```

In [24]:
```
#vraag 10
#we mogen aannemend dat die nips normaal is
#verdeelt door de reprensentatie
```

In [25]:
```
#vraag 11
import pandas as pd
tips = tips.fillna(tips.mean())

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
tips[['total_bill', 'time']] = scaler.fit_transform(tips[['total_bill', 'time'

from sklearn.model_selection import train_test_split

train_data, test_data = train_test_split(tips, test_size=0.25, random_state=42

from sklearn.linear_model import LinearRegression

model = LinearRegression()

X_train = train_data[['total_bill', 'time', 'size']]
y_train = train_data['tip']

model.fit(X_train, y_train)

from sklearn.metrics import mean_absolute_error, r2_score

y_pred = model.predict(X_train)
mae = mean_absolute_error(y_train, y_pred)
r2 = r2_score(y_train, y_pred)

print(f"Training MAE: {mae:.3f}, R-Squared: {r2:.3f}")
X_test = test_data[['total_bill', 'time', 'size']]
y_pred = model.predict(X_test)
```