import libraries and dataset

```
import pandas as pd
```

```
df= pd.read_csv("/content/dataset_2.csv")
df
```

	Name	Trips Completed	Accept Rate	Supply Hours	Rating
0	Abdul	1	100%	3	4.8
1	Abraham	12	83%	5	4.7
2	Adelina	1	100%	2	4.7
3	Akilah	1	100%	2	4.9
4	Alec	21	76%	11	5.0
...	...	...	...	...	...
114	Virgen	4	100%	6	4.5
115	Yang	7	71%	2	4.5
116	Yessenia	8	88%	5	5.0
117	Yukiko	9	78%	6	4.5
118	Zoe	21	86%	9	4.9

119 rows × 5 columns

```
df.head(5)
```

	Name	Trips Completed	Accept Rate	Supply Hours	Rating
0	Abdul	1	100%	3	4.8
1	Abraham	12	83%	5	4.7
2	Adelina	1	100%	2	4.7
3	Akilah	1	100%	2	4.9
4	Alec	21	76%	11	5.0

see the shape of data

```
df.shape
```

```
(119, 5)
```

see the size of data

```
df.size
```

```
595
```

see data types of data

```
df.dtypes
```

```
Name                 object
Trips Completed       int64
Accept Rate          object
Supply Hours          int64
Rating              float64
dtype: object
```

change the dtype of rating column from float to integer

```
df['Rating'] = df['Rating'].astype(int)
df.dtypes
```

```
    Name               object
    Trips Completed     int64
    Accept Rate        object
    Supply Hours        int64
    Rating              int64
    dtype: object
```

change the dtype of accept rate column from object to integer by removing % sign first then covert it into integer value

```
df['Accept Rate'] = df['Accept Rate'].str.rstrip('%').astype(int) # rstrip removes the % sign from each value,
## in the 'Accept Rate' column, and then astype(int) converts the resulting strings to integers.
```

***check the dtypes, all have been changed now ***

```
df.dtypes
```

```
    Name               object
    Trips Completed     int64
    Accept Rate         int64
    Supply Hours        int64
    Rating              int64
    dtype: object
```

Do label encoding to change the "Name" (object) column into integer

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
```

now name column has been changed into integer

```
df['Name'] = label_encoder.fit_transform(df['Name'])
df.dtypes
```

```
    Name               int64
    Trips Completed    int64
    Accept Rate        int64
    Supply Hours       int64
    Rating             int64
    dtype: object
```

Scenario 1: It is going to be a huge Saturday and there will need to be many more cars on the road than last week. In order to get drivers to go online, we're assessing the following two bonus options in terms of cost: • Option 1: $50foreachdriverthatisonlineatleast8hours, accepts90 \bullet Option2$:4/trip for all drivers who complete 12 trips, and have a 4.7 or better rating. Using the dataset provided and given Scenario 1, provide answers to the questions below:

1. How much would the total bonus payout be with Option 1?
2. How much would the total bonus payout be with Option 2?
3. How many drivers would qualify for a bonus under Option 1 but not under Option 2?
4. What percentages of drivers online completed less than 10 trips, had an acceptance rate of less than 90%, and had a rating of 4.7 or higher?

• Option 1: $50 for each driver that is online at least 8 hours, accepts 90% of requests, completes 10 trips, and has a rating of 4.7 or better during the time frame

Ques 1:How much would the total bonus payout be with Option 1? bold text

***Ques 1:How much would the total bonus payout be with Option 1? ***

```
c_for_opt1= (df['Supply Hours'] >= 8) & \
           (df['Accept Rate'] >= 0.9) & \
           (df['Trips Completed'] >= 10) & \
           (df['Rating'] >= 4.7)
```

```
qualified_drivers = df[c_for_opt1]
qualified_drivers.count()
```

```
Name               14
Trips Completed    14
Accept Rate        14
Supply Hours       14
Rating             14
dtype: int64
```

```
bonus_per_driver = 50
total_bonus_payout = len(qualified_drivers)*50
```

```
total_bonus_payout
```

```
700
```

- Option 2: $4/trip for all drivers who complete 12 trips, and have a 4.7 or better rating.

   2. How much would the total bonus payout be with Option 2?

2.How much would the total bonus payout be with Option 2?

```
#c_for_option2 = (df['Trips Completed']>=12) & \
               # (df['Rating']>=4.7)
```

```
c_for_option2 = df[(df['Trips Completed'] >= 12) & (df['Rating'] >= 4.7)]
```

```
qual_driver=c_for_option2
```

```
qual_driver.count()
```

```
Name               15
Trips Completed    15
Accept Rate        15
Supply Hours       15
Rating             15
dtype: int64
```

```
qual_driver['bonus']= qual_driver['Trips Completed']*4
```

```
<ipython-input-22-a3a213c9d8b4>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  qual_driver['bonus']= qual_driver['Trips Completed']*4
```

```
total_bonus_payout = qual_driver['bonus'].sum()
total_bonus_payout
```

```
1032
```

- Option 2: $4/trip for all drivers who complete 12 trips, and have a 4.7 or better rating.

   3. How many drivers would qualify for a bonus under Option 1 but not under Option 2?

3.How many drivers would qualify for a bonus under Option 1 but not under Option 2?

```
qualified_drivers
```

	Name	Trips Completed	Accept Rate	Supply Hours	Rating
4	4	21	76	11	5
12	12	14	100	8	5
23	23	14	100	9	5
32	32	12	100	9	5
37	37	15	100	10	5
56	56	24	88	11	5
57	57	13	100	9	5
70	70	21	95	10	5
75	75	23	100	10	5
76	76	16	90	10	5
87	87	18	89	11	5
96	96	18	44	11	5
102	102	21	67	11	5
104	104	15	80	9	5

```
qualified_d_df = pd.concat([qualified_drivers, qual_driver])
qualified_d_df
```

	Name	Trips Completed	Accept Rate	Supply Hours	Rating	bonus
4	4	21	76	11	5	NaN
12	12	14	100	8	5	NaN
23	23	14	100	9	5	NaN
32	32	12	100	9	5	NaN
37	37	15	100	10	5	NaN
56	56	24	88	11	5	NaN
57	57	13	100	9	5	NaN
70	70	21	95	10	5	NaN
75	75	23	100	10	5	NaN
76	76	16	90	10	5	NaN
87	87	18	89	11	5	NaN
96	96	18	44	11	5	NaN
102	102	21	67	11	5	NaN
104	104	15	80	9	5	NaN
4	4	21	76	11	5	84.0
10	10	13	54	6	5	52.0
12	12	14	100	8	5	56.0
23	23	14	100	9	5	56.0
32	32	12	100	9	5	48.0
37	37	15	100	10	5	60.0
56	56	24	88	11	5	96.0
57	57	13	100	9	5	52.0
70	70	21	95	10	5	84.0
75	75	23	100	10	5	92.0
76	76	16	90	10	5	64.0
87	87	18	89	11	5	72.0
96	96	18	44	11	5	72.0
102	102	21	67	11	5	84.0
104	104	15	80	9	5	60.0

```
qualified_count = qualified_drivers['Name'].count()
qual_count = qual_driver['Name'].count()
```

```
result = qual_count-qualified_count
result
```

```
1
```

4.What percentages of drivers online completed less than 10 trips, had an acceptance rate of less than 90%, and had a rating of 4.7 or higher?

per% of drivers= qualifying drivers/total no of online drivers

```
#qualifying_drivers= len(df[df['Supply Hours']> 0])
#qualifying_drivers
```

```
df['Supply Hours'].count()
```

```
119
```

```
total_no_of_online_drivers=df['Supply Hours'].count()
total_no_of_online_drivers

     119
```

```
total_no_of_online_drivers= len(df['Supply Hours'])
total_no_of_online_drivers

     119
```

```
qualifying_drivers= [(df['Trips Completed'] < 10) & \
           (df['Accept Rate'] >= 0.90) & \
           (df['Rating'] >= 4.7)]
```

```
#qualifying_drivers
```

```
quality_driver=qualifying_drivers
quality_driver

     [0      False
      1      False
      2      False
      3      False
      4      False
             ...
      114    False
      115    False
      116     True
      117    False
      118    False
      Length: 119, dtype: bool]
```

```
#percentage = (len(qualifying_drivers) / len(df['Supply Hours'])) * 100
#percentage
```

```
percentage = (len(qualifying_drivers) / total_no_of_online_drivers) * 100
percentage

     0.8403361344537815
```

Scenario 2: A taxi driver currently generates $200 perdayinfares(beforeexpenses), workssixdaysaweek, takesthreeweeksoff, andhasthefollowingexpenses: \bullet Gas-$ 200 per week • Insurance - $400 permonth \bullet Vehiclerent(bytheweek)-$500 The driver doesn't pay gas and rent expenses on off weeks. Now, let's assume that the same driver would buy a Town Car and partner with Uber. If he does, his gas expenses would go up by 5%, his insurance expense would decrease by 20%, and he would no longer be renting a vehicle. However, he would need to buy a car. The driver would still take three weeks off per year. Given Scenario 2, provide answers to the questions below:

5. How much money (after expenses) does the taxi driver make per year without partnering with Uber?
6. You are convincing the same driver above to buy a Town Car and partner with Uber. Assuming the new car is 40,000 USD, how much would the driver's gross fares need to increase per week to fully pay for the car in year 1 and maintain the same yearly profit margin as before?

5.How much money (after expenses) does the taxi driver make per year without partnering with Uber?

```
#yearly_income = income_per_day * days_per_week * week_per_year # calculate the annual income

income_per_day= 200
days_per_week= 6
week_per_year= 52
yearly_income = income_per_day * days_per_week * week_per_year # calculate the annual income
yearly_income = 200 * 6 * 52
yearly_income


     62400
```

```
# yearly expenses:

gas_per_week=200
```

```
gas_per_week= 200
insurance_per_month= 500
vehicle_rent_per_week= 400
weeks_per_year= 52


gas_per_year= gas_per_week * week_per_year
gas_per_year= 200 * 52
insurance_per_year= (insurance_per_month *12) /4
insurance_per_year= 400 * 12/4
vehicle_rent_per_rent= vehicle_rent_per_week * weeks_per_year
vehicle_rent_per_week= 500 * 52
total_expenses = gas_per_year + insurance_per_year +vehicle_rent_per_rent
total_expenses
```

```
     32400.0
```

```
annual_net_income = yearly_income - total_expenses
annual_net_income
```

```
     30000.0
```

6.You are convincing the same driver above to buy a Town Car and partner with Uber. Assuming the new car is 40,000 USD, how much would the driver's gross fares need to increase per week to fully pay for the car in year 1 and maintain the same yearly profit margin as before?

```
daily_fares_before_expense = 200 # Daily fares before expenses in USD
days_per_week = 6      # no of working days
week_per_year = 49   # Number of working weeks in a year (52 weeks - 3 weeks off)
gas_expense_per_week = 200  ## Gas expenses per week
insurance_expense_per_month = 400 ## Insurance expenses per month
vehicle_rent_per_week = 500  # Vehicle rent per week
initial_car_cost = 40000  #Initial investment for the Town Car
```

```
# calculate total expenses per week before and after:
total_expense_per_week_before = gas_expense_per_week + (insurance_expense_per_month * 12/52)+ vehicle_rent_per_week
new_gas_expense_per_week = 1.05 * gas_expense_per_week # gas exp increased by 5%
new_insurance_per_week = (1 - 0.2) * (insurance_expense_per_month * 12/52) # insurance decrease by 20%
additional_expense_per_week = initial_car_cost / week_per_year
total_expense_per_week_after = new_gas_expense_per_week + new_insurance_per_week + additional_expense_per_week
```

```
# calculate increase in gross fares per week:
increase_in_gross_fares_per_week = total_expense_per_week_after - total_expense_per_week_before


increase_in_gross_fares_per_week
```

```
     307.8649921507065
```