


```
import pandas as pd
df_pokemon=pd.read_csv('Pokemon.csv')
```

1. Find the pokemon names for which Legendary=True

```
df_legend=df_pokemon[df_pokemon['Legendary']==True]
df_legend
```



	#	Name	Type1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Ge
156	144	Articuno	Ice	Flying	580	90	85	100	95	125	85	
157	145	Zapdos	Electric	Flying	580	90	90	85	125	90	100	
158	146	Moltres	Fire	Flying	580	90	100	90	125	85	90	
162	150	Mewtwo	Psychic	NaN	680	106	110	90	154	90	130	
163	150	MewtwoMega Mewtwo X	Psychic	Fighting	780	106	190	100	154	100	130	
...	
795	719	Diancie	Rock	Fairy	600	50	100	150	100	150	50	
796	719	DiancieMega Diancie	Rock	Fairy	700	50	160	110	160	110	110	
797	720	HoopaHoopa Confined	Psychic	Ghost	600	80	110	60	150	130	70	
		HoopaHoopa										

2. Get pokemon name from user input and then print details about the pokemon.

```
name=input("Enter the pokemon name:")
df_pokemon[df_pokemon['Name']==name]
```

Enter the pokemon name:Charmeleon

#	Name	Type1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation

3. Find pokemon with maximum Total. And if tie then check Legendary=True if still its a tie check Generation=3

```
max_total=df_pokemon[df_pokemon['Total']==max(df_pokemon['Total'])]
```

```
max_total[max_total['Generation']==3]
```

#	Name	Type1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Gen

4. Pokemon details which has Generation=6

```
df_pokemon[df_pokemon['Generation']==6]
```

#	Name	Type1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Gen
718	650	Chespin	Grass	NaN	313	56	61	65	48	45	38
719	651	Quilladin	Grass	NaN	405	61	78	95	56	58	57
720	652	Chesnaught	Grass	Fighting	530	88	107	122	74	75	64
721	653	Fennekin	Fire	NaN	307	40	45	40	62	60	60
722	654	Braixen	Fire	NaN	409	59	59	58	90	70	73
...
795	719	Diancie	Rock	Fairy	600	50	100	150	100	150	50
796	719	DiancieMega Diancie	Rock	Fairy	700	50	160	110	160	110	110
797	720	HoopaHoopa Confined	Psychic	Ghost	600	80	110	60	150	130	70
798	720	HoopaHoopa Unbound	Psychic	Dark	680	80	160	60	170	130	80

```
df_car=pd.read_csv('car.csv')
```

```
df_car.head()
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	NI
0	BMW	Series 1 M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	
1	BMW	Series 1	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	
2	BMW	Series 1	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	
3	BMW	Series 1	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	
4	BMW	Series 1	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	

▼ 1. Most expensive car for each car company.

```
df=df_car[['Make', 'MSRP']].groupby('Make').max()
df.head()
```

	MSRP
Make	
Acura	156000
Alfa Romeo	68400
Aston Martin	320695
Audi	199900
BMW	141200

▼ 2. Most expensive convertible vehicle for each company.

```
df_convertible=df_car[df_car['Vehicle Style']=='Convertible']
df_convertible.head()
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	N
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	
4	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	
6	BMW	1 Series	2012	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	
8	BMW	1 Series	2012	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	
		1		premium					

```
df=df_convertible[['Make','MSRP']].groupby('Make').max()  
df.head()
```

	MSRP
Make	
Alfa Romeo	65900
Aston Martin	320695
Audi	199600
BMW	136900
Bentley	363000

▼ 3. List all cars that are automatic.

```
df_auto_car=df_car[df_car['Transmission Type']=='AUTOMATIC']  
df_auto_car.head()
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors
19	Audi	100	1992	regular unleaded	172.0	6.0	AUTOMATIC	all wheel drive	4.0
23	Audi	100	1993	regular	172.0	6.0	AUTOMATIC	all wheel drive	4.0

▼ 4. Premium Cars of each car company.

```
df_premium=df_car[df_car['Engine Fuel Type']=='premium unleaded (required)']
df_premium.head()
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors
0	BMW	Series 1 M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0
1	BMW	Series 1	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0
2	BMW	Series 1	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0
3	BMW	Series 1	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0
4	BMW	Series 1	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0

```
df_car['Engine Fuel Type'].unique()

array(['premium unleaded (required)', 'regular unleaded',
      'premium unleaded (recommended)', 'flex-fuel (unleaded/E85)',
      'diesel', 'electric',
      'flex-fuel (premium unleaded recommended/E85)', 'natural gas',
      'flex-fuel (premium unleaded required/E85)',
      'flex-fuel (unleaded/natural gas)', nan], dtype=object)
```

▼ 5. Average price of cars for each company.

```
df=df_car[['Make','MSRP']].groupby('Make').mean()  
df.head()
```

MSRP	
Make	
Acura	34887.587302
Alfa Romeo	61600.000000
Aston Martin	197910.376344
Audi	53452.112805
BMW	61546.763473