```python
from datetime import date
from datetime import datetime

departtime = datetime.fromtimestamp(1651104000)
con_no = int(input("Enter Consignment number: "))              # datetime MODULE USE CASE

if (con_no < 3000):                                  # IF CONSIGNMENT LESS THAN 3000, PACKAGE IS STILL IN TRANSIT
    print("Package Departed on", departtime)
else:
    print("Sorry Your order is still in transit.")
```

In [51]:

```
import pandas as pd
f1 = pd.DataFrame(
    {
        "Driver Number" : [33,11,16,55,44,63],
        "Driver": ["Max Verstappen", "Sergio Perez", "Charles Leclerc", "Carlos Sainz Jr.",
        "Team": [
            "RedBull Racing",
            "RedBull Racing",
            "Scuderia Ferrari",                    #CREATED A CUSTOM DATAFRAME IN PANDAS
            "Scuderia Ferrari",
            "Mercedes AMG",
            "Mercedes AMG"
        ],
        "Nationality" : ["Netherlands", "Mexico","Monaco", "Spain","Britain", "Britain"],
        "Age": [24,30,24,26,38,25],
    }
)
f1
f1["Age"].min()  #GIVES MINMUM AGE OF THE DRIVER AMONG ALL DRIVERS
f1.head(2) #DISPLAYS ONLY 2 ROWS
f1.sort_values("Age") #DRIVERS ARE SORTED BY THEIR AGES
f1.where(f1["Driver"] == "Max Verstappen")  # FINDS DRIVER NAMED "Max Verstappen"
f1.loc[f1["Age"]<35] #DISPLAYS DRIVERS UNDER 35
```

Out[51]:

	Driver Number	Driver	Team	Nationality	Age
0	33	Max Verstappen	RedBull Racing	Netherlands	24
1	11	Sergio Perez	RedBull Racing	Mexico	30
2	16	Charles Leclerc	Scuderia Ferrari	Monaco	24
3	55	Carlos Sainz Jr.	Scuderia Ferrari	Spain	26
5	63	George Russell	Mercedes AMG	Britain	25

In [ ]:

```
import pandas as pd #IMPORT PANDAS
f1 = pd.read_csv('ParticipantData_10230136787177318441.csv') #READING FROM CSV
```

```
f1.loc[f1["teamId"] == "Toro Rosso"]
```

	pilot_index	aiControlled	driverId	teamId	maxRPM	idleRPM
0	0	1	Pierre Gasly	Toro Rosso	13000	3499
16	16	1	Daniil Kvyat	Toro Rosso	13000	3499

```
f1.where(f1["idleRPM"]>4000)
```

	pilot_index	aiControlled	driverId	teamId	maxRPM	idleRPM
0	NaN	NaN	NaN	NaN	NaN	NaN
1	1.0	1.0	Charles Leclerc	Ferrari	13000.0	4300.0
2	NaN	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN
4	4.0	1.0	Sebastian Vettel	Ferrari	13000.0	4300.0
5	NaN	NaN	NaN	NaN	NaN	NaN
6	6.0	1.0	Lance Stroll	Racing Point	13000.0	4300.0
7	NaN	NaN	NaN	NaN	NaN	NaN
8	8.0	1.0	Antonio Giovinazzi	Alfa Romeo	13000.0	4300.0
9	9.0	1.0	Kevin Magnussen	Haas	13000.0	4300.0
10	NaN	NaN	NaN	NaN	NaN	NaN
11	11.0	1.0	Nicholas Latifi	Williams	13000.0	4300.0
12	12.0	1.0	Lewis Hamilton	Mercedes	13000.0	4300.0
13	13.0	1.0	Romain Grosjean	Haas	13000.0	4300.0
14	14.0	1.0	George Russell	Williams	13000.0	4300.0
15	15.0	1.0	Sergio Perez	Racing Point	13000.0	4300.0
16	NaN	NaN	NaN	NaN	NaN	NaN
17	17.0	1.0	Kimi Räikkönen	Alfa Romeo	13000.0	4300.0
18	NaN	NaN	NaN	NaN	NaN	NaN
19	19.0	0.0	Valtteri Bottas	Mercedes	13000.0	4300.0

Double-click (or enter) to edit

```
f1.sort_values("idleRPM").head(5)
```

	pilot_index	aiControlled	driverId	teamId	maxRPM	idleRPM
0	0	1	Pierre Gasly	Toro Rosso	13000	3499
2	2	1	Max Verstappen	Red Bull Racing	13000	3499
16	16	1	Daniil Kvyat	Toro Rosso	13000	3499
10	10	1	Alexander Albon	Red Bull Racing	13000	3499
3	3	1	Lando Norris	McLaren	13000	3799

+ Code + Text

✓ 0s completed at 12:15 PM

In [6]:

```
#LIST COMPREHENSION
names = ["Rajasekhara Reddy", "Raghava Reddy", "Jagan Mohan Reddy",
         "Balayya Chowdary", "Ramayya Chowdary", "Surya Patil", "Rahul Patil", "Sundeep Pat

newlist = [x for x in names if "Patil" in x]       # PARSING THROUGH THE LIST

print(newlist)
```

```
['Surya Patil', 'Rahul Patil', 'Sundeep Patil']
```

In [20]:

```

```

In [16]:

```
class car:
  def __init__(self, name, enginename):
    self.carname = name
    self.engine = enginename
                                        # SINGLE INHERITANCE
  def printname(self):
    print(self.carname, self.engine)


x = car("Ferrari Italia", "4.5 L V8",)
x.printname()
```

```
Ferrari Italia 4.5 L V8
```

In [22]:

```
class Employees():

   def Name(self):
    print ("Employee Name: Sai")

class salary(Employees):
   def Salary(self):
       print ("Salary: 5000")    #MULTILEVEL INHERITANCE

class Designation(salary):
   def desig(self):
       print ("Designation: Trainee Data Engineer")

call = Designation()
call.Name()
call.Salary()
call.desig()
```

```
Employee Name: Sai
Salary: 5000
Designation: Trainee Data Engineer
```

In [ ]: