In [1]:
```
#IMPORTING REQUIRED LIBRARY

import os
import pandas as pd
import numpy as np
import matplotlib as plt
import datetime as dt
import seaborn as sns
```

In [ ]:
```

```

In [160]:
```
import import_ipynb
import matplotlib.pyplot as plt1
```

In [3]:
```
%matplotlib inline
import sklearn
```

In [4]:
```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
```

In [5]:
```
from sklearn.ensemble import RandomForestRegressor
```

In [6]:
```
import statsmodels.api as sm
```

In [7]:
```
from sklearn.neighbors import KNeighborsRegressor
```

In [8]:
```
#SETTING WORKING DIRECTORY
os.chdir("E:/data science and machine learning/Employee Absentism project 2/Pytho
```

In [9]:
```
os.getcwd()
```

Out[9]:
```
'E:\\data science and machine learning\\Employee Absentism project 2\\Python'
```

In [10]:
```
#GETTING THE FILE FROM HDD
edf=pd.read_csv("Absent.csv",sep=',')
```

In [11]:
```
type(edf)
```

Out[11]:
```
pandas.core.frame.DataFrame
```

In [12]:
```
edf.columns
```

Out[12]:
```
Index(['Unnamed: 0', 'ID', 'Absent_reason', 'Absent_month', 'Absent_day',
       'Seasons', 'Transport_expense', 'Distance', 'Service_time', 'Age',
       'Work_load', 'Hit_target', 'Discipline_failure', 'Education', 'Son',
       'Social_drinker', 'Social_smoker', 'Pet', 'Weight', 'Height',
       'Body_mass', 'Absent_hours'],
      dtype='object')
```

In [13]:
```
edf.dtypes
```

Out[13]:
```
Unnamed: 0           int64
ID                   int64
Absent_reason        int64
Absent_month         int64
Absent_day           int64
Seasons              int64
Transport_expense    int64
Distance             int64
Service_time         int64
Age                  int64
Work_load            int64
Hit_target           int64
Discipline_failure   int64
Education            int64
Son                  int64
Social_drinker       int64
Social_smoker        int64
Pet                  int64
Weight               int64
Height               int64
Body_mass            int64
Absent_hours         int64
dtype: object
```

In [14]:
```
missing_val=pd.DataFrame(edf.isnull().sum())
```

In [15]:
```
missing_val
```

Out[15]:

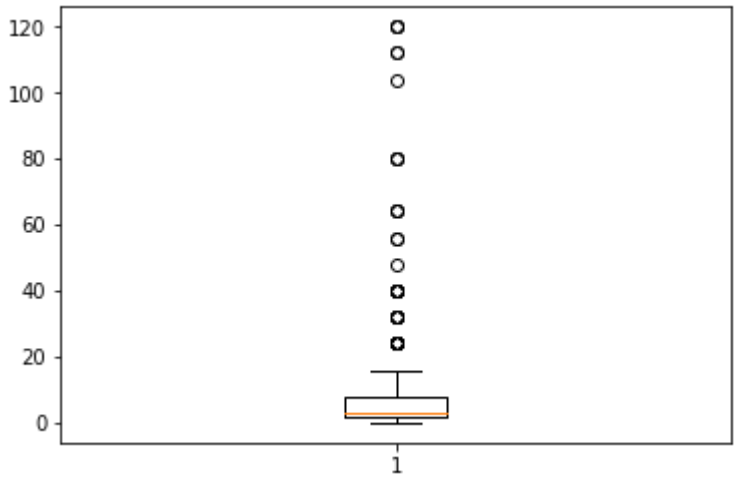
	0
Unnamed: 0	0
ID	0
Absent_reason	0
Absent_month	0
Absent_day	0
Seasons	0
Transport_expense	0
Distance	0
Service_time	0
Age	0
Work_load	0
Hit_target	0
Discipline_failure	0
Education	0
Son	0
Social_drinker	0
Social_smoker	0
Pet	0
Weight	0
Height	0
Body_mass	0
Absent_hours	0

In [16]:
```
del edf['Unnamed: 0']
```

In [17]:
```
edf.columns
```

Out[17]:
```
Index(['ID', 'Absent_reason', 'Absent_month', 'Absent_day', 'Seasons',
       'Transport_expense', 'Distance', 'Service_time', 'Age', 'Work_load',
       'Hit_target', 'Discipline_failure', 'Education', 'Son',
       'Social_drinker', 'Social_smoker', 'Pet', 'Weight', 'Height',
       'Body_mass', 'Absent_hours'],
      dtype='object')
```

In [18]:
```
#BOX PLOT TO CHECK OUTLIERS OF EVERY VARIABLE
plt.boxplot(edf['Absent_hours'])
cnames=['ID', 'Absent_reason', 'Absent_month', 'Absent_day', 'Seasons',
       'Transport_expense', 'Distance', 'Service_time', 'Age', 'Work_load',
       'Hit_target', 'Discipline_failure', 'Education', 'Son',
       'Social_drinker', 'Social_smoker', 'Pet', 'Weight', 'Height',
       'Body_mass']
```

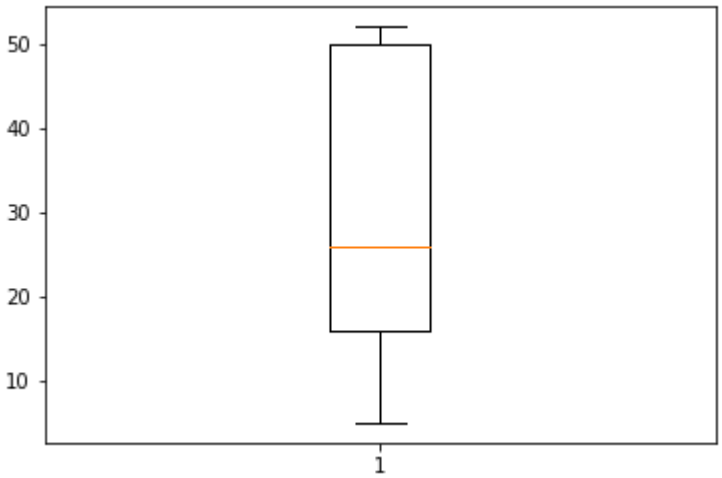


In [47]:
```
edfc=edf.copy()
```

In [ ]:
```

```

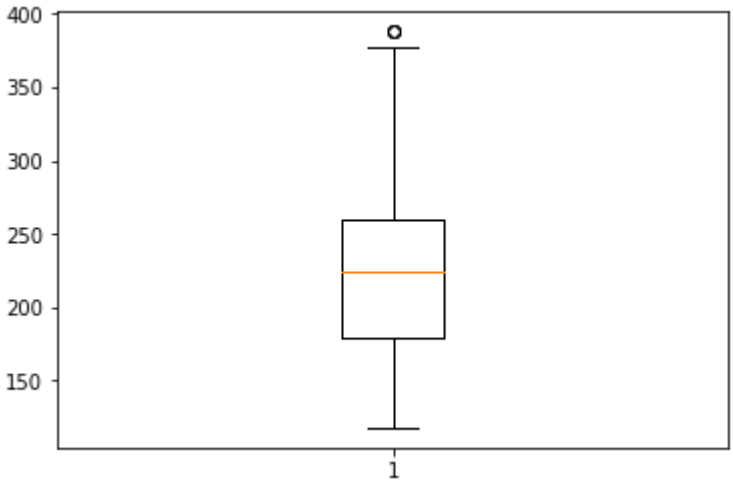
In [32]:
```
plt.boxplot(edf['Distance'])
```

Out[32]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1930b32ee10>,
  <matplotlib.lines.Line2D at 0x1930b338470>],
 'caps': [<matplotlib.lines.Line2D at 0x1930b3387b8>,
  <matplotlib.lines.Line2D at 0x1930b338b00>],
 'boxes': [<matplotlib.lines.Line2D at 0x1930b32ecc0>],
 'medians': [<matplotlib.lines.Line2D at 0x1930b338e48>],
 'fliers': [<matplotlib.lines.Line2D at 0x1930b338f28>],
 'means': []}
```



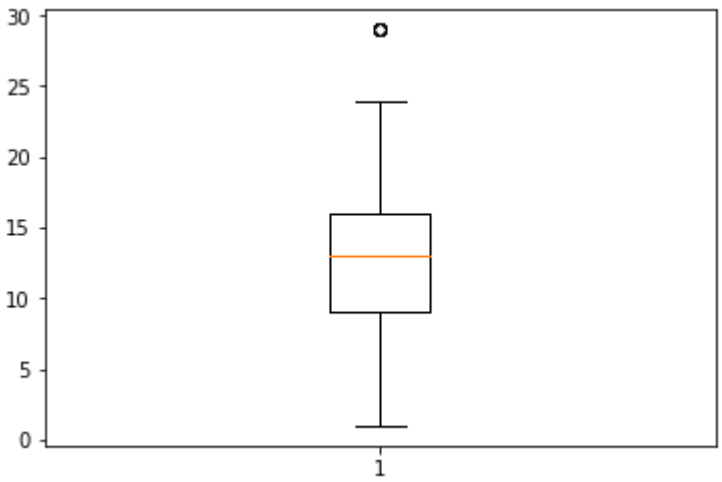
In [33]:
```
plt.boxplot(edf['Transport_expense'])
```

Out[33]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1930b38e358>,
  <matplotlib.lines.Line2D at 0x1930b38e6a0>],
 'caps': [<matplotlib.lines.Line2D at 0x1930b38e9e8>,
  <matplotlib.lines.Line2D at 0x1930b38ed30>],
 'boxes': [<matplotlib.lines.Line2D at 0x1930b384ef0>],
 'medians': [<matplotlib.lines.Line2D at 0x1930b38ee10>],
 'fliers': [<matplotlib.lines.Line2D at 0x1930b398400>],
 'means': []}
```



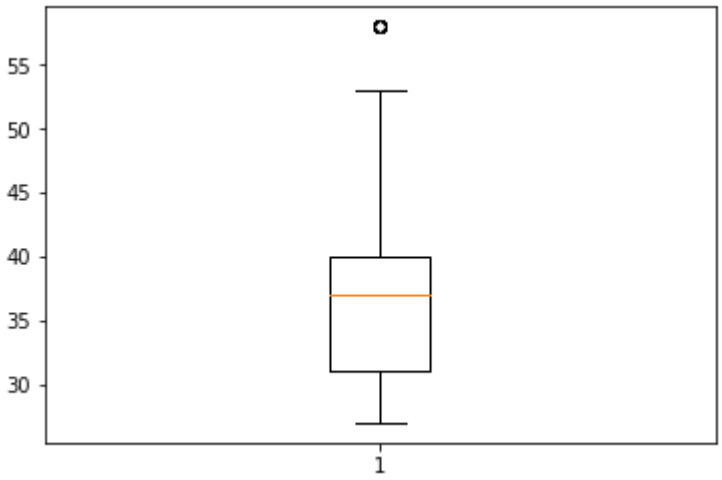
In [34]:
```
plt.boxplot(edf['Service_time'])
```

Out[34]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1930b3decf8>,
  <matplotlib.lines.Line2D at 0x1930b3dedd8>],
 'caps': [<matplotlib.lines.Line2D at 0x1930b3e93c8>,
  <matplotlib.lines.Line2D at 0x1930b3e9710>],
 'boxes': [<matplotlib.lines.Line2D at 0x1930b3de8d0>],
 'medians': [<matplotlib.lines.Line2D at 0x1930b3e9a58>],
 'fliers': [<matplotlib.lines.Line2D at 0x1930b3e9da0>],
 'means': []}
```



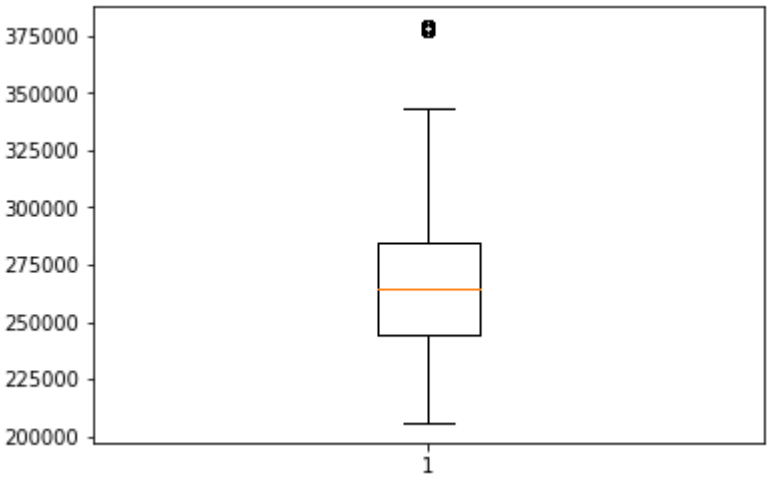
In [35]:
```
plt.boxplot(edf['Age'])
```

Out[35]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1930b43b828>,
  <matplotlib.lines.Line2D at 0x1930b43bb70>],
 'caps': [<matplotlib.lines.Line2D at 0x1930b43beb8>,
  <matplotlib.lines.Line2D at 0x1930b43bf98>],
 'boxes': [<matplotlib.lines.Line2D at 0x1930b43b400>],
 'medians': [<matplotlib.lines.Line2D at 0x1930b447588>],
 'fliers': [<matplotlib.lines.Line2D at 0x1930b4478d0>],
 'means': []}
```



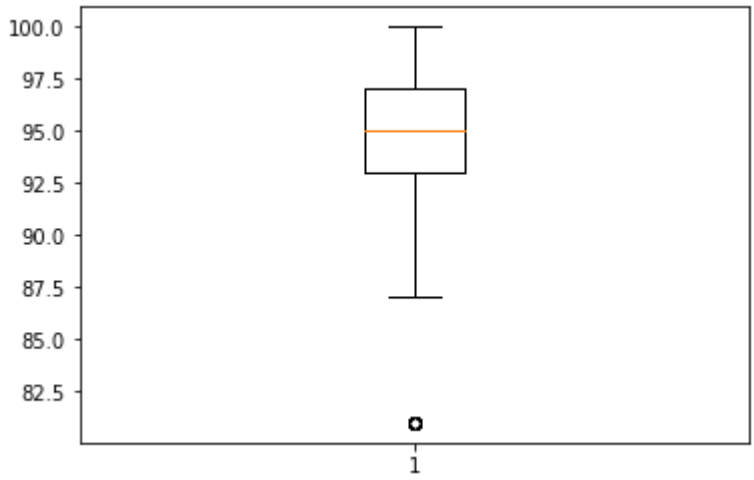
In [36]:
```
plt.boxplot(edf['Age'])
```

Out[36]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1930b48fd30>,
  <matplotlib.lines.Line2D at 0x1930b496390>],
 'caps': [<matplotlib.lines.Line2D at 0x1930b4966d8>,
  <matplotlib.lines.Line2D at 0x1930b496a20>],
 'boxes': [<matplotlib.lines.Line2D at 0x1930b48fbe0>],
 'medians': [<matplotlib.lines.Line2D at 0x1930b496d68>],
 'fliers': [<matplotlib.lines.Line2D at 0x1930b496e48>],
 'means': []}
```



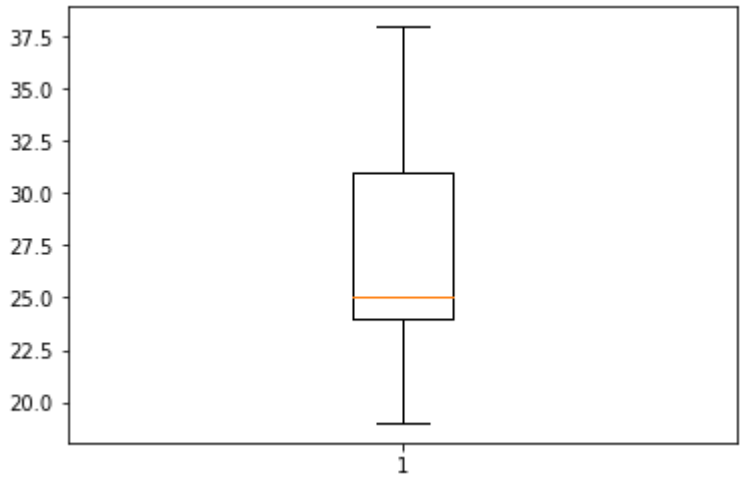
In [37]:
```
plt.boxplot(edf['Hit_target'])
```

Out[37]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1930c4c7390>,
  <matplotlib.lines.Line2D at 0x1930c4c76d8>],
 'caps': [<matplotlib.lines.Line2D at 0x1930c4c7a20>,
  <matplotlib.lines.Line2D at 0x1930c4c7d68>],
 'boxes': [<matplotlib.lines.Line2D at 0x1930c4bcf28>],
 'medians': [<matplotlib.lines.Line2D at 0x1930c4c7e48>],
 'fliers': [<matplotlib.lines.Line2D at 0x1930c4cf438>],
 'means': []}
```

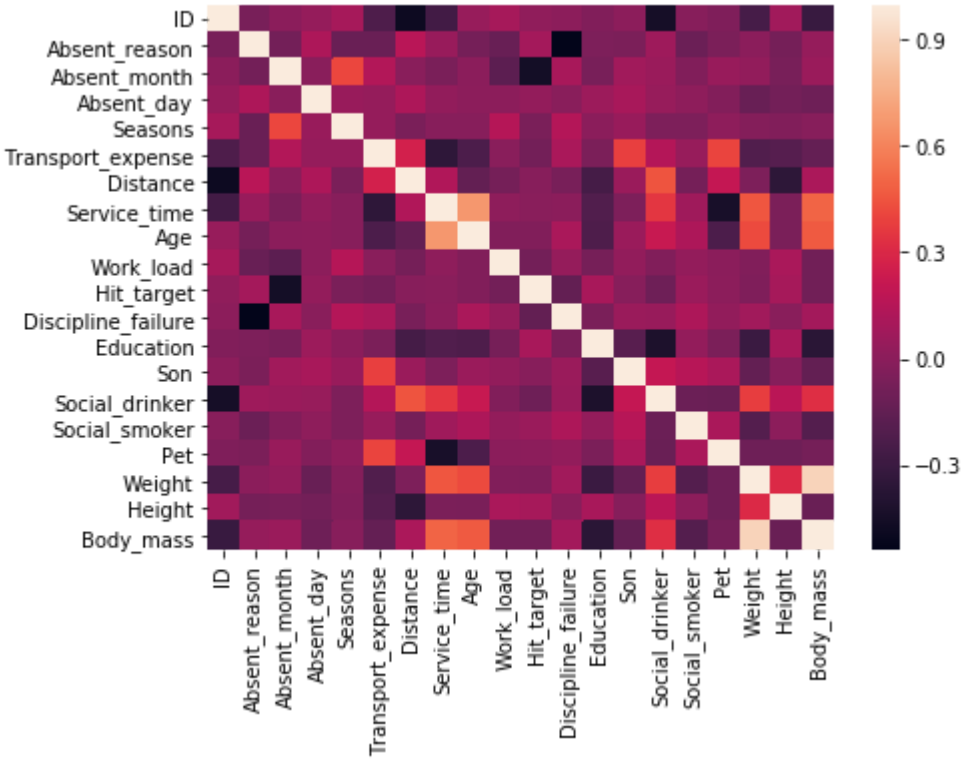


In [38]:
```
plt.boxplot(edf['Body_mass'])
```

Out[38]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1930c522438>,
  <matplotlib.lines.Line2D at 0x1930c522780>],
 'caps': [<matplotlib.lines.Line2D at 0x1930c522ac8>,
  <matplotlib.lines.Line2D at 0x1930c522e10>],
 'boxes': [<matplotlib.lines.Line2D at 0x1930c517fd0>],
 'medians': [<matplotlib.lines.Line2D at 0x1930c522ef0>],
 'fliers': [<matplotlib.lines.Line2D at 0x1930c52a4e0>],
 'means': []}
```



In [39]:
```
#FEATURE SELECTION
edf_corr=edf.loc[:,cnames]

f,ax=plt.subplots(figsize=(7,5))
corr=edf_corr.corr()
ax = sns.heatmap(corr)
```



In [40]:
```
del edf['Height']
```

In [41]:
```
del edf['Weight']
```

In [42]:
```
edf.columns
```

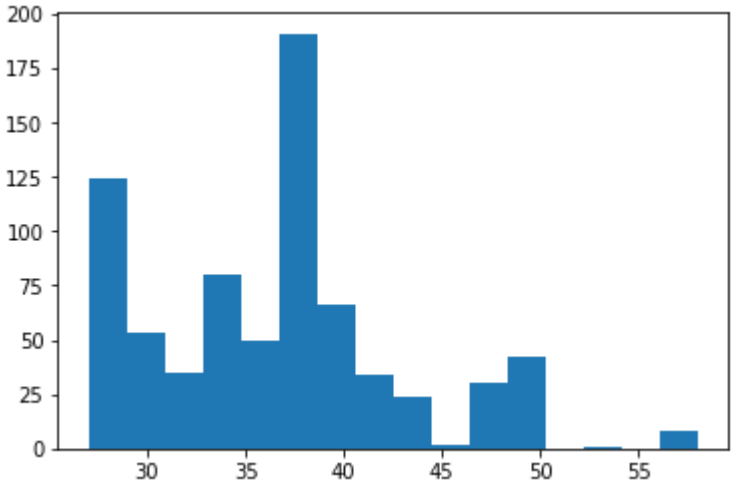
Out[42]:
```
Index(['ID', 'Absent_reason', 'Absent_month', 'Absent_day', 'Seasons',
       'Transport_expense', 'Distance', 'Service_time', 'Age', 'Work_load',
       'Hit_target', 'Discipline_failure', 'Education', 'Son',
       'Social_drinker', 'Social_smoker', 'Pet', 'Body_mass', 'Absent_hours'],
      dtype='object')
```

In [43]:
```
cnames1=["Transport_expense","Distance","Service_time","Work_load","Hit_target"]
edf
edf.dtypes
```

Out[43]:
```
ID                   int64
Absent_reason        int64
Absent_month         int64
Absent_day           int64
Seasons              int64
Transport_expense    int64
Distance             int64
Service_time         int64
Age                  int64
Work_load            int64
Hit_target           int64
Discipline_failure   int64
Education            int64
Son                  int64
Social_drinker       int64
Social_smoker        int64
Pet                  int64
Body_mass            int64
Absent_hours         int64
dtype: object
```

In [44]:
```
plt.hist(edf['Age'],bins='auto')
```

Out[44]:
```
(array([124.,  53.,  35.,  80.,  50., 191.,  66.,  34.,  24.,   2.,  30.,
         42.,   0.,   1.,   0.,   8.]),
 array([27.    , 28.9375, 30.875 , 32.8125, 34.75  , 36.6875, 38.625 ,
        40.5625, 42.5   , 44.4375, 46.375 , 48.3125, 50.25  , 52.1875,
        54.125 , 56.0625, 58.    ]),
 <a list of 16 Patch objects>)
```



In [45]:
```
#NORMALIZATION
for j in cnames1:
    print(j)

    edf[j]=(edf[j]-min(edf[j]))/(max(edf[j])-min(edf[j]))
```

```
Transport_expense
Distance
Service_time
Work_load
Hit_target
```

In [46]:
```
edf
```

Out[46]:

	ID	Absent_reason	Absent_month	Absent_day	Seasons	Transport_expense	Distance	Servi
0	11	26	7	3	1	0.633333	0.659574	0
1	36	0	7	3	1	0.000000	0.170213	0
2	3	23	7	4	1	0.225926	0.978723	0
3	7	7	7	5	1	0.596296	0.000000	0
4	11	23	7	5	1	0.633333	0.659574	0
5	3	23	7	6	1	0.225926	0.978723	0
6	10	22	7	6	1	0.900000	1.000000	0
7	20	23	7	6	1	0.525926	0.957447	0
8	14	19	7	2	1	0.137037	0.148936	0
9	1	22	7	2	1	0.433333	0.127660	0
10	20	1	7	2	1	0.525926	0.957447	0
11	20	1	7	3	1	0.525926	0.957447	0
12	20	11	7	4	1	0.525926	0.957447	0
13	3	11	7	4	1	0.225926	0.978723	0
14	3	23	7	4	1	0.225926	0.978723	0
15	24	14	7	6	1	0.474074	0.425532	0
16	3	23	7	6	1	0.225926	0.978723	0
17	3	21	7	2	1	0.225926	0.978723	0
18	6	11	7	5	1	0.262963	0.510638	0
19	33	23	8	4	1	0.481481	0.425532	0
20	18	10	8	4	1	0.785185	0.234043	0
21	3	11	8	2	1	0.225926	0.978723	0
22	10	13	8	2	1	0.900000	1.000000	0
23	20	28	8	6	1	0.525926	0.957447	0
24	11	18	8	2	1	0.633333	0.659574	0
25	10	25	8	2	1	0.900000	1.000000	0
26	11	23	8	3	1	0.633333	0.659574	0
27	30	28	8	4	1	0.144444	0.468085	0
28	11	18	8	4	1	0.633333	0.659574	0
29	3	23	8	6	1	0.225926	0.978723	0
...	...	...	...	...	...	...	...	
710	23	22	6	5	3	0.962963	0.936170	0
711	36	19	6	5	3	0.000000	0.170213	0
712	12	19	6	6	3	0.425926	0.978723	0

	ID	Absent_reason	Absent_month	Absent_day	Seasons	Transport_expense	Distance	Servi
713	22	27	6	6	3	0.225926	0.446809	0
714	2	0	6	2	3	0.433333	0.510638	0
715	21	0	6	2	3	0.555556	0.127660	0
716	36	19	6	5	3	0.000000	0.170213	0
717	22	13	6	5	3	0.225926	0.446809	0
718	15	28	6	5	3	0.640741	0.553191	0
719	22	13	6	2	1	0.225926	0.446809	0
720	34	25	6	2	1	0.000000	0.106383	0
721	12	22	6	5	1	0.425926	0.978723	0
722	34	8	6	6	1	0.000000	0.106383	0
723	34	10	6	4	1	0.000000	0.106383	0
724	12	22	6	4	1	0.425926	0.978723	0
725	5	26	7	4	1	0.433333	0.319149	0
726	12	19	7	6	1	0.425926	0.978723	0
727	9	6	7	2	1	0.407407	0.191489	0
728	34	28	7	2	1	0.000000	0.106383	0
729	9	6	7	3	1	0.407407	0.191489	0
730	6	22	7	3	1	0.262963	0.510638	0
731	34	23	7	4	1	0.000000	0.106383	0
732	10	22	7	4	1	0.900000	1.000000	0
733	28	22	7	4	1	0.396296	0.446809	0
734	13	13	7	2	1	0.929630	0.255319	0
735	11	14	7	3	1	0.633333	0.659574	0
736	1	11	7	3	1	0.433333	0.127660	0
737	4	0	0	3	1	0.000000	0.191489	0
738	8	0	0	4	2	0.418519	0.638298	0
739	35	0	0	6	3	0.225926	0.851064	0

740 rows × 19 columns

In [50]:
```
#SAMPLING
train, test = train_test_split(edf, test_size=0.3)
```

In [74]:
```
#LINEAR REGRESSION

model = sm.OLS(train.iloc[:,18], train.iloc[:,0:17]).fit()
```

In [75]:
```
predictions_LR = model.predict(test.iloc[:,0:17])
```

In [76]:
```
predictions_LR
```

Out[76]:
```
738    19.471482
314    10.593398
348     2.759214
86      8.259459
20      9.196415
698     3.709145
343    10.747855
408    -0.277520
643     1.325051
383     9.568194
104     9.406173
94     10.734849
324    14.572628
527     6.499649
607    -0.332034
204    12.471706
472     0.948933
518     1.998713
433    19.619864
542     8.717855
153     2.173732
369     1.494707
395    10.008298
76      0.034338
426    11.676728
321    13.702604
205     5.304397
731     3.338670
625     1.325051
446    -1.307939
         ...
293     4.947677
454    11.532294
710     6.179836
452     2.984170
276     2.526846
182    13.003362
699     7.190584
213    10.862257
123     1.774067
344     7.924052
735    12.842434
150     0.254968
251     8.943526
124     0.863300
506     3.157045
465    -0.673626
36      7.204477
641     2.235819
54      5.138674
485    10.830447
57      3.314002
64      8.694793
685    14.637304
427    21.566310
```

```
200     8.461998
439    21.599765
385     0.986191
95      0.734533
320     9.642466
192    11.129822
Length: 222, dtype: float64
```

In [126]:
```
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from math import sqrt

rms_LR = sqrt(mean_squared_error(test['Absent_hours'],predictions_LR))
mae_LR = mean_absolute_error(test.iloc[:,18],predictions_LR)
```

In [127]:
```
rms_LR
mae_LR
```

Out[127]:
```
5.23319247555218
```

In [56]:
```
#RMSE_LR=7.52
#MAE_LR=5.23
```

In [145]:
```
#DECISIONS TREE
fit_DT = DecisionTreeRegressor(max_depth=3).fit(train.iloc[:,0:17], train.iloc[:
```

In [146]:
```
predictions_DT = fit_DT.predict(test.iloc[:,0:17])
```

In [147]:
```
predictions_DT
```

Out[147]:
```
array([ 13.58      ,  10.20491803,   3.33913043,  16.        ,
        10.20491803,   3.175     ,   1.5       ,   3.33913043,
         3.33913043,   3.175     ,  10.20491803,   4.15384615,
         7.63636364,   1.68292683,   3.33913043,  13.58      ,
         3.33913043,   1.68292683,  13.58      ,  10.20491803,
         2.7       ,   3.33913043,   7.25      ,   3.33913043,
         3.175     ,   3.175     ,   1.68292683,   3.33913043,
         3.33913043,   0.15      ,   3.175     ,   7.63636364,
        10.20491803,  32.33333333,  32.33333333,   3.175     ,
         1.68292683,   3.33913043,  10.20491803,  10.20491803,
         3.33913043,  32.33333333,   2.14285714,   3.33913043,
         3.175     ,   4.15384615,   7.25      ,   2.14285714,
         3.33913043,  10.20491803,   1.        ,   3.33913043,
         7.25      ,   4.15384615,   3.33913043,  10.20491803,
        10.20491803,   3.33913043,   2.14285714,   1.68292683,
         1.68292683,   3.33913043,  10.20491803,  10.20491803,
         1.5       ,  13.58      ,   3.33913043,   2.7       ,
         3.33913043,   3.175     ,   3.33913043,   0.15      ,
         3.        ,   3.33913043,   3.33913043,  13.58      ,
         3.33913043,   1.68292683,  13.58      ,   1.68292683,
         4.15384615,   3.33913043,   3.33913043,   7.25      ,
         3.33913043,   7.25      ,   3.33913043,   7.63636364,
         8.        ,   2.7       ,   3.33913043,   3.33913043,
         3.33913043,   7.25      ,  10.20491803,  10.20491803,
         7.25      ,  32.33333333,  25.27272727,   4.15384615,
         3.175     ,   0.15      ,   7.63636364,   3.33913043,
         3.33913043,   3.175     ,  10.20491803,   3.175     ,
         2.7       ,  13.58      ,   2.14285714,   3.175     ,
         3.175     ,   7.25      ,  10.20491803,   1.68292683,
         7.63636364, 104.        ,  13.58      ,   3.175     ,
        10.20491803,   2.14285714,   2.14285714,   3.175     ,
         1.68292683,   3.33913043,   3.33913043,   7.25      ,
        13.58      ,   3.33913043,  10.20491803,  10.20491803,
        10.20491803,  10.20491803,   2.14285714,   3.33913043,
         3.33913043,   4.15384615,  24.        ,   3.175     ,
       104.        ,   8.        ,   3.33913043,  10.20491803,
         3.33913043,   3.33913043,   3.33913043,  10.20491803,
         3.33913043,  10.20491803,   3.175     ,  32.33333333,
        13.58      ,  10.20491803, 112.        ,   1.68292683,
        10.20491803,   3.175     ,  13.58      ,   3.33913043,
         7.25      ,  10.20491803,   2.14285714,  13.58      ,
        10.20491803,   1.68292683,   3.33913043,   0.15      ,
         3.33913043,   1.68292683,  10.20491803,   3.175     ,
        10.20491803,  13.58      ,  13.58      ,   3.175     ,
         3.33913043,   2.14285714,   4.15384615,   3.33913043,
         7.25      ,  10.20491803,  13.58      ,   0.15      ,
        10.20491803,   3.33913043,  13.58      ,  10.20491803,
         3.175     ,  13.58      ,  10.20491803,   3.33913043,
         0.15      ,   1.5       ,   7.25      ,   3.33913043,
        13.58      ,  13.58      ,   2.14285714,  32.33333333,
         1.68292683,   4.15384615,  13.58      ,   2.7       ,
        13.58      ,   1.68292683,   7.25      ,   3.33913043,
         3.175     ,   3.33913043,  13.58      ,   7.63636364,
         3.33913043,   0.15      ,   1.5       ,  13.58      ,
```

```
       13.58      ,  10.20491803,   3.33913043,   1.68292683,
        4.15384615,   1.68292683])
```

In [149]:
```
rms_DT = sqrt(mean_squared_error(test['Absent_hours'],predictions_DT))
mae_DT = mean_absolute_error(test.iloc[:,18],predictions_DT)
```

In [150]:
```
#RMSE_DT=8.78
#MAE_DT=4.69

mae_DT
rms_DT
```

Out[150]:
```
13.385731756783144
```

In [100]:
```
#RANDOM FOREST
RF_model = RandomForestRegressor(n_estimators = 700).fit(train.iloc[:,0:17], tra
```

In [101]:
```
predictions_RF = RF_model.predict(test.iloc[:,0:17])
```

In [102]:
```
predictions_RF
```

Out[102]:
```
array([ 4.29428571, 42.47142857,  3.64769048, 11.90785714, 29.16714286,
        3.54261905,  2.8717619 ,  5.32230952,  2.08416667,  3.50266667,
       44.19871429,  4.10369048,  6.71418367,  4.43971429,  2.43205102,
       20.03      ,  4.01114626,  1.98952381, 16.29571429,  8.68142857,
        1.9195102 ,  3.14508844,  8.18833333,  2.62488095,  2.82947619,
        3.20894558,  2.01007143,  3.80919048,  2.08416667,  0.66285714,
        9.42890476,  6.59630952,  6.83      , 16.11142857, 33.835     ,
        3.87864286,  2.01      ,  2.41625   , 11.06285714,  9.63142857,
        3.3227381 , 23.22785714,  2.355     ,  2.57619048,  5.85964286,
        4.29607143,  8.15285714,  2.85044218,  2.56787415,  8.11428571,
        5.43057143,  2.12210714,  8.4       ,  4.93116667,  2.02592857,
        3.28142857, 12.47464286,  3.19371429,  2.9027415 ,  2.56861905,
        1.85114286,  2.49333333,  9.70142857,  5.77428571,  2.8997619 ,
       14.84821429,  2.84109524,  2.57238095,  3.18385714,  2.27028571,
        2.40358503,  3.10428571,  4.64714286,  2.02592857,  3.18385714,
       10.92      ,  2.22712245,  2.14714286, 26.33285714,  2.59664286,
        4.12178571,  4.31552381,  3.54933333,  7.21571429,  2.76511905,
        9.34666667,  4.50214286,  6.3887415 , 60.93428571,  2.70161905,
        3.54263605,  2.79180952,  2.06995238,  7.64285714, 21.58857143,
        9.54285714,  8.67761905, 18.90857143, 57.87328571,  3.6934881 ,
        5.05238095,  4.49892857,  7.11440476,  3.69928571,  3.01014286,
        3.19857143,  7.92      ,  3.39214286,  1.66619048, 14.22392857,
        4.36140476,  2.59321429,  4.61877551,  6.24      ,  6.08857143,
        2.58214286,  6.96393939, 44.19      ,  6.76571429,  3.87487075,
        7.71478571,  2.2727381 ,  2.70559524,  3.24557143,  2.04      ,
        2.86678571,  3.2467381 ,  8.24428571,  7.36857143,  3.94990476,
        3.77285714, 19.48      ,  9.43571429, 26.24285714,  1.5725    ,
        4.08195238,  2.08416667,  3.78107143, 42.83428571,  4.46114286,
       46.56571429,  7.66714286,  2.02797619,  7.38      ,  3.04116667,
        2.3502381 ,  2.79928571, 10.45714286,  3.81085714, 17.28142857,
        3.23619048, 28.92571429,  5.49285714,  8.22142857, 45.52928571,
        1.76042517, 29.85428571,  2.88407143, 16.75285714,  4.05487415,
       10.56583333, 13.91      ,  1.6769932 , 15.49571429, 19.92285714,
        1.9979932 ,  2.03619048,  0.73428571,  3.94990476,  1.25087755,
       14.60428571,  3.14440816, 18.8       , 25.12071429,  9.02      ,
        1.90414286,  2.79093311,  3.20215986,  3.67184354,  1.3112619 ,
        7.4447619 ,  7.61      , 16.52857143,  2.87178571,  3.99428571,
        2.12319048,  8.67428571, 18.73857143,  2.8352381 , 13.09735714,
        5.00571429,  2.00214286,  4.40178571,  2.77185714,  7.78214286,
        9.41176536,  6.83428571, 11.78285714,  2.35516667, 44.67285714,
        1.85114286,  4.56609524, 13.96      ,  2.14285714,  3.22      ,
        1.97980952,  7.92214286,  5.72278571,  3.74892857,  2.06642857,
        4.45142857,  7.17952381,  4.42978571,  4.39821429,  1.9602381 ,
       31.43571429, 21.70571429,  9.67142857,  3.01333333,  1.97580952,
        4.08471429,  1.78428571])
```

In [131]:
```
rms_RF = sqrt(mean_squared_error(test['Absent_hours'],predictions_RF))
mae_RF = mean_absolute_error(test.iloc[:,18],predictions_RF)
```

In [132]:
```
#RMSE_RF=10.26
#MAE_RF=4.97
rms_RF
mae_RF
```

Out[132]:
```
4.975010388403246
```

In [120]:
```
#KNN
KNN_model=KNeighborsRegressor(n_neighbors=9).fit(train.iloc[:,0:17], train.iloc[
```

In [121]:
```
KNN_Predictions=KNN_model.predict(test.iloc[:,0:17])
```

In [122]:
```
KNN_Predictions
```

Out[122]:
```
array([ 4.        , 14.22222222,  3.44444444,  7.66666667,  4.77777778,
        2.        ,  1.22222222,  3.33333333,  2.55555556,  3.55555556,
       11.33333333,  1.44444444,  4.44444444,  6.33333333,  2.33333333,
       24.88888889,  6.55555556,  2.22222222, 22.22222222, 11.22222222,
        1.66666667,  1.55555556,  4.44444444,  2.33333333,  2.11111111,
        3.66666667,  2.44444444,  4.55555556,  2.55555556, 15.44444444,
        3.33333333,  5.44444444, 17.44444444, 24.88888889, 28.44444444,
        6.66666667,  2.22222222,  2.33333333,  4.22222222, 18.44444444,
        3.44444444, 19.55555556,  2.88888889,  2.66666667,  5.44444444,
        4.55555556,  3.77777778,  2.33333333,  3.88888889,  4.33333333,
        2.66666667,  2.22222222,  4.        ,  4.66666667,  2.33333333,
       15.44444444, 15.33333333,  3.77777778,  3.        ,  3.11111111,
        1.55555556,  3.77777778,  5.11111111,  3.11111111,  4.22222222,
       19.        ,  2.88888889,  1.77777778,  3.55555556,  1.33333333,
        3.33333333, 11.55555556,  4.66666667,  2.33333333,  3.55555556,
        3.11111111,  2.44444444,  2.22222222, 15.66666667,  2.22222222,
        4.55555556,  3.55555556,  3.55555556,  5.        ,  3.22222222,
       12.88888889,  3.55555556,  4.22222222, 26.66666667,  2.44444444,
        4.        ,  2.44444444,  2.33333333,  5.66666667,  5.        ,
       14.66666667,  4.11111111, 26.66666667, 29.66666667,  4.77777778,
        3.55555556,  3.11111111,  7.44444444,  4.66666667,  2.22222222,
        3.22222222, 14.66666667,  2.22222222,  1.55555556, 17.33333333,
        6.66666667,  1.33333333,  4.88888889,  5.55555556, 18.44444444,
        2.44444444,  5.77777778, 17.77777778,  3.55555556, 10.88888889,
        3.88888889,  3.88888889,  2.22222222,  2.22222222,  3.11111111,
        2.55555556,  4.        , 11.77777778,  7.11111111, 10.77777778,
        2.11111111,  6.66666667, 14.44444444,  7.55555556,  3.88888889,
        4.55555556,  2.55555556,  4.66666667, 28.44444444,  3.44444444,
       16.88888889,  4.        ,  2.22222222, 10.44444444,  2.66666667,
        2.11111111,  2.        ,  3.11111111, 10.88888889,  5.44444444,
        3.11111111,  7.11111111,  3.11111111,  9.22222222, 31.66666667,
        2.11111111, 14.22222222,  1.33333333, 16.33333333,  3.44444444,
        4.77777778, 13.55555556,  3.88888889, 19.        , 17.11111111,
        2.44444444,  2.44444444, 20.44444444, 10.77777778,  1.55555556,
        7.55555556,  5.77777778, 15.88888889, 14.66666667, 12.66666667,
        1.77777778,  1.77777778,  3.88888889,  4.11111111,  1.55555556,
        5.66666667,  6.33333333,  9.66666667, 11.11111111, 18.        ,
        1.66666667, 14.33333333,  9.55555556,  2.44444444,  4.77777778,
       18.        ,  2.33333333,  2.55555556,  4.22222222, 13.66666667,
        7.11111111, 13.77777778, 28.44444444,  2.77777778, 15.55555556,
        1.55555556,  4.66666667, 25.77777778,  1.77777778,  3.55555556,
        1.55555556,  5.55555556,  3.55555556,  3.55555556,  2.44444444,
       19.55555556,  5.77777778,  2.55555556,  2.55555556,  4.77777778,
       26.66666667, 13.        , 16.77777778,  2.33333333,  2.22222222,
        4.66666667,  4.66666667])
```

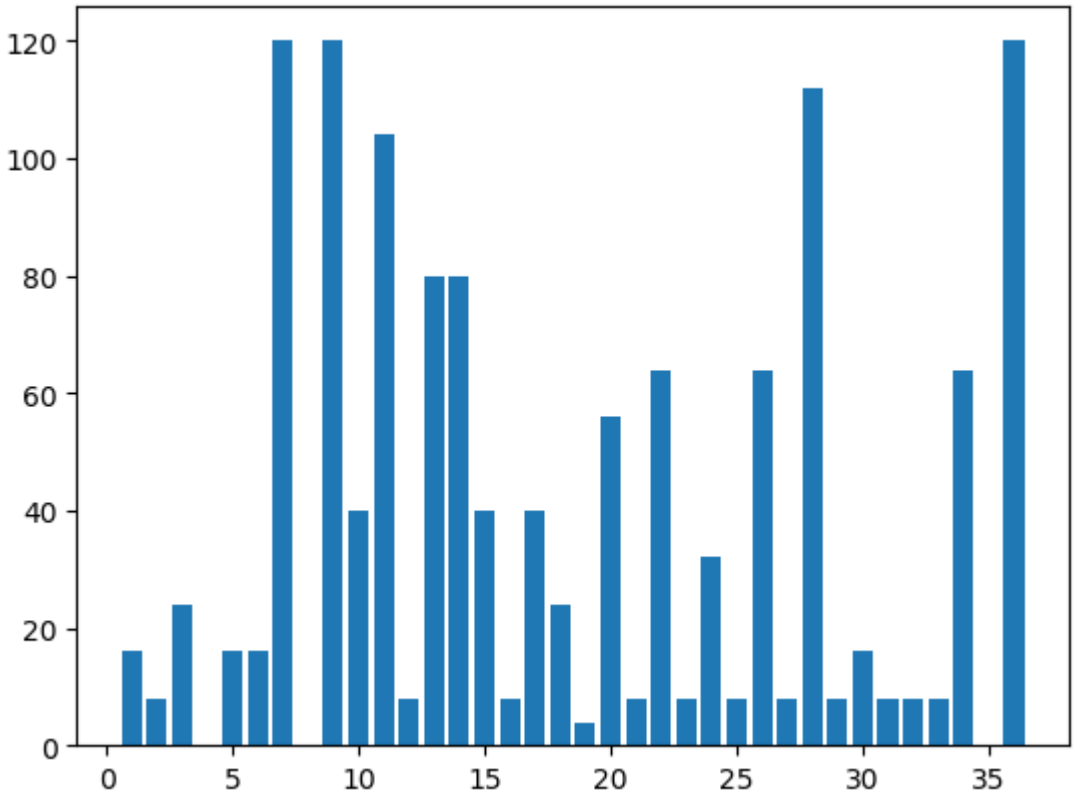
In [133]:
```
rms_KNN = sqrt(mean_squared_error(test['Absent_hours'],KNN_Predictions))
mae_KNN = mean_absolute_error(test.iloc[:,18],KNN_Predictions)
```

In [162]:
```
#RMSE_KNN=7.84
#MAE_KNN=4.77
rms_KNN
mae_KNN
```

Out[162]:
```
4.776776776776776
```

In [188]:
```
plt1.rcdefaults()
plt1.bar(edf['ID'],edf['Absent_hours'])
```

Out[188]:
```
<BarContainer object of 740 artists>
```



In [175]:
```
edf['ID'].nunique()
```

Out[175]:
```
36
```

In [177]:
```
edf['ID'].value_counts()
```

Out[177]:
```
3     113
28     76
34     55
22     46
20     42
11     40
15     37
36     34
24     30
14     29
33     24
10     24
1      23
17     20
5      19
18     16
13     15
25     10
6       8
9       8
23      8
27      7
30      7
12      7
7       6
2       6
29      5
26      5
32      5
31      3
19      3
21      3
16      2
8       2
35      1
4       1
Name: ID, dtype: int64
```

In [178]:
```
edf['ID'].describe()
```

Out[178]:
```
count    740.000000
mean      18.017568
std       11.021247
min        1.000000
25%        9.000000
50%       18.000000
75%       28.000000
max       36.000000
Name: ID, dtype: float64
```

In [218]:
```
edfID=edf.groupby('ID',).sum()[['Absent_hours']]
edfID
```

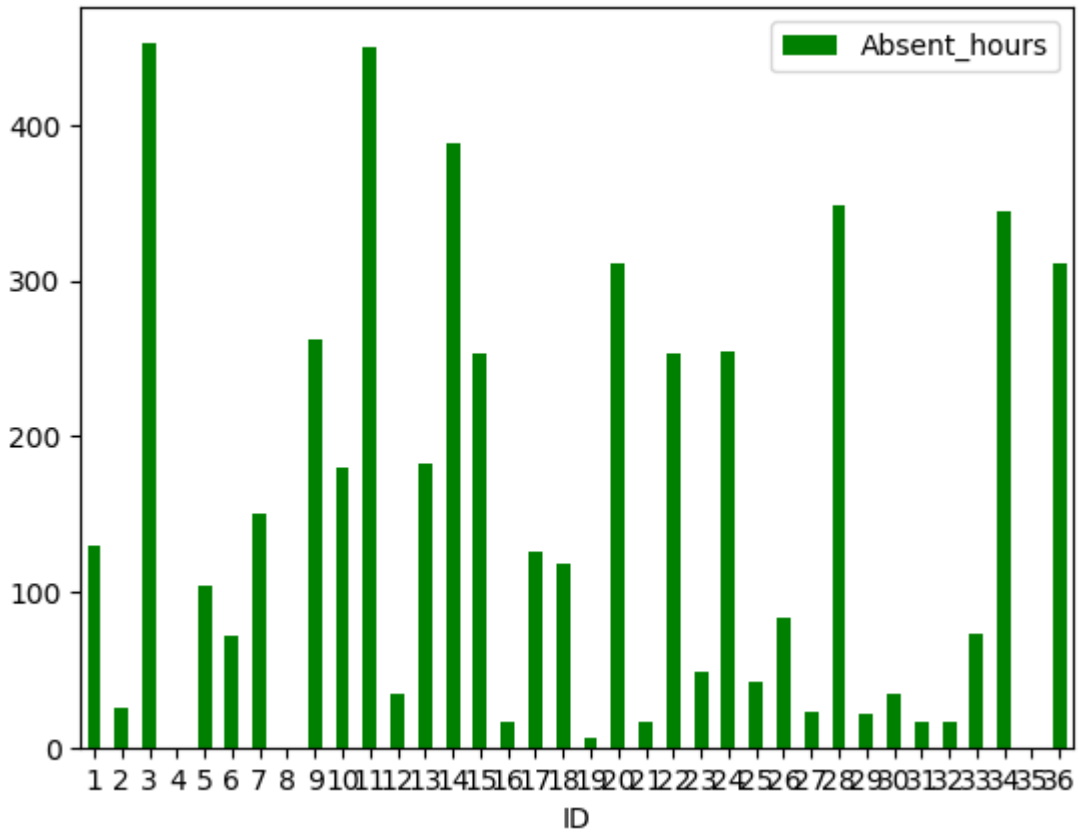
Out[218]:

ID	Absent_hours
1	129
2	25
3	453
4	0
5	104
6	72
7	150
8	0
9	262
10	180
11	450
12	34
13	183
14	388
15	253
16	16
17	126
18	118
19	6
20	311
21	16
22	253
23	48
24	254
25	42
26	83
27	23
28	348
29	21
30	34
31	16
32	16

ID	Absent_hours
33	73
34	344
35	0
36	311

In [226]:
```
edfID.plot.bar(rot=0,color='green')
```

Out[226]: <matplotlib.axes._subplots.AxesSubplot at 0x193104f6a90>



In [213]:
```
edfAge=edf.groupby('Age',as_index=True).sum()[['Absent_hours']]
edfAge
```

Out[213]:

Age	Absent_hours
27	23
28	646
29	34
30	253
31	217
32	48
33	538
34	388
36	359
37	473
38	453
39	150
40	379
41	275
43	187
46	16
47	73
48	25
49	16
50	327
53	0
58	262

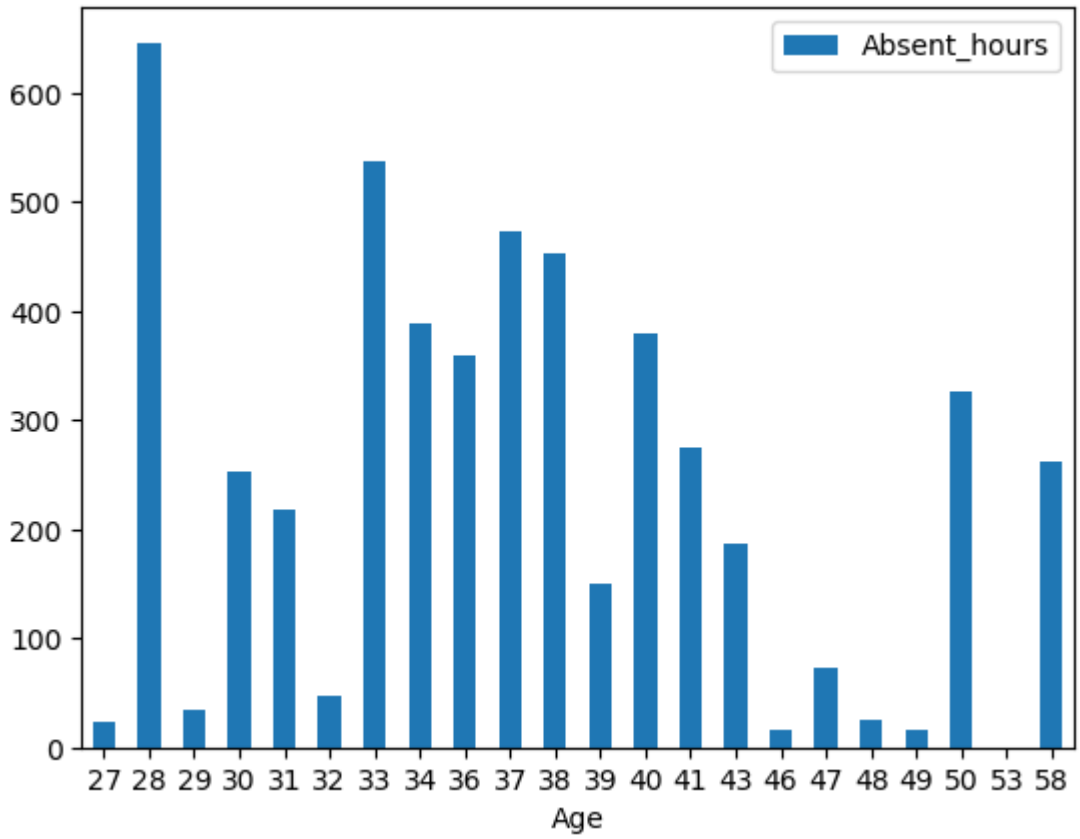
In [ ]:

In [ ]:

In [ ]:

In [217]:
```
edfAge.plot.bar(rot=0)
```

Out[217]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1930fba7160>
```



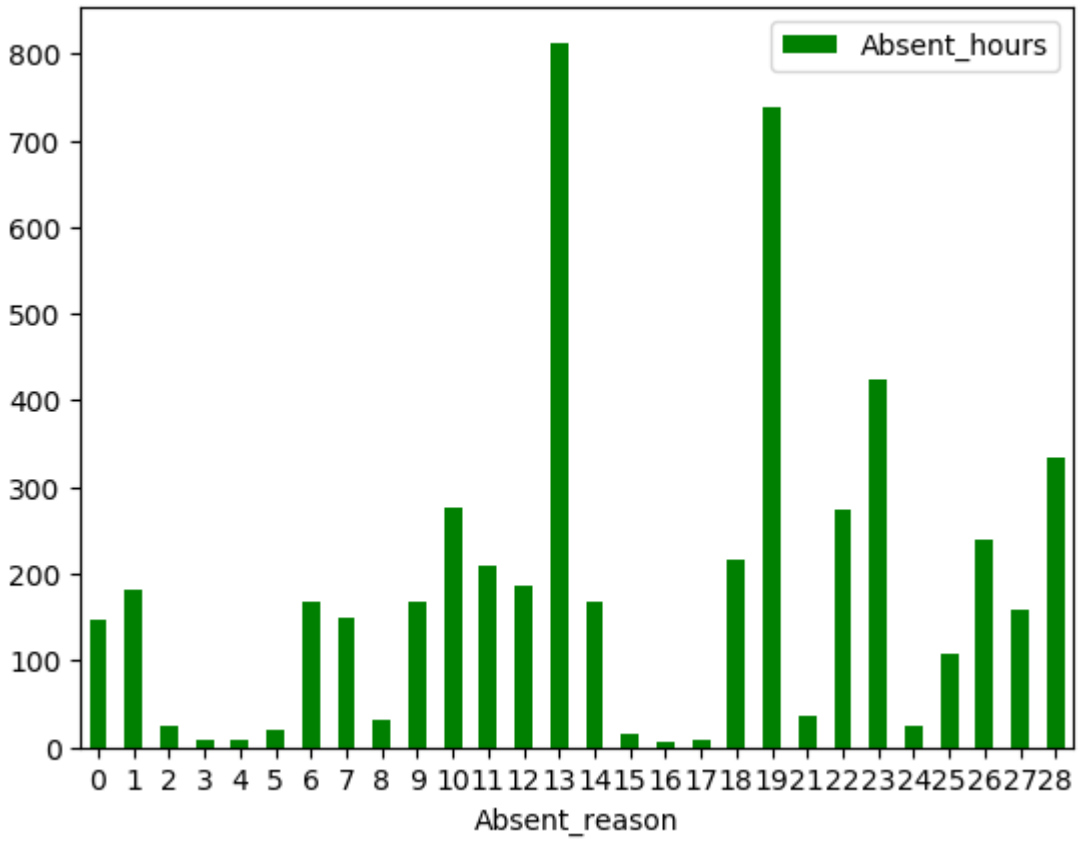
In [228]:
```
edfAbs=edf.groupby('Absent_reason',as_index=True).sum()[['Absent_hours']]
edfAbs
```

Out[228]:

Absent_reason	Absent_hours
0	147
1	182
2	24
3	8
4	9
5	19
6	167
7	150
8	32
9	168
10	276
11	209
12	187
13	813
14	167
15	16
16	6
17	8
18	217
19	737
21	35
22	275
23	425
24	24
25	108
26	240
27	158
28	335

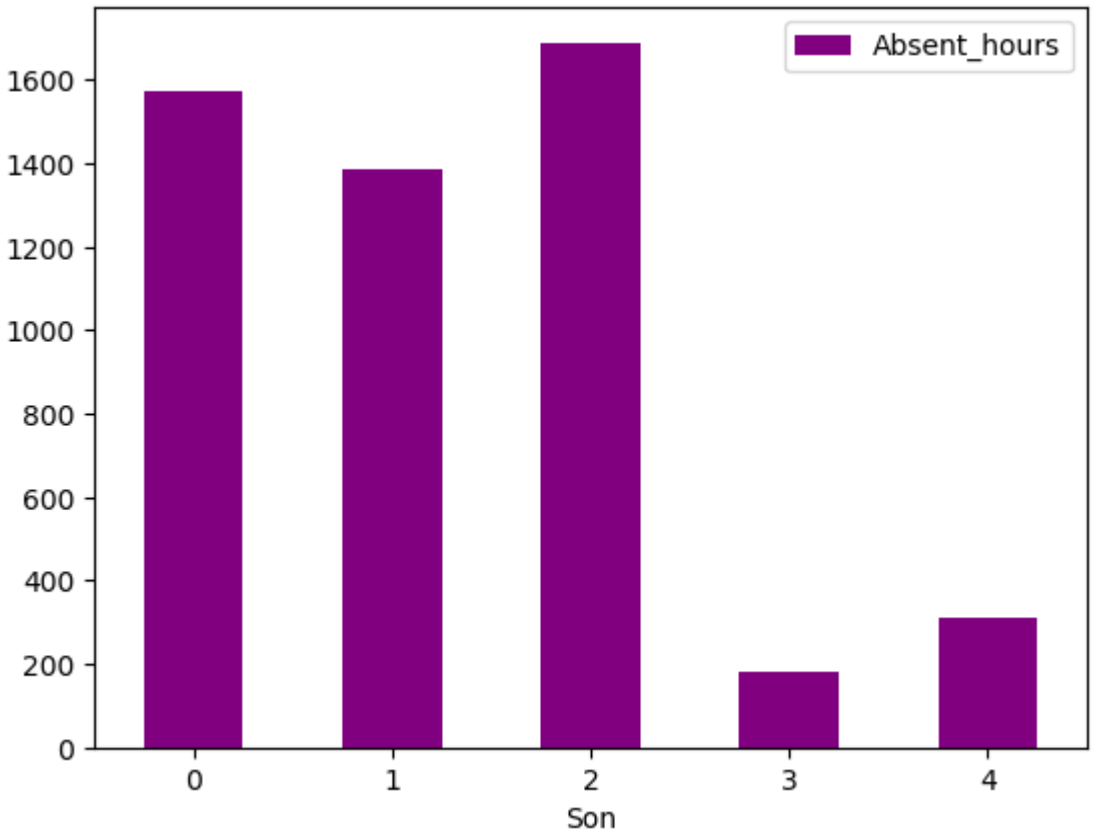
In [229]:
```
edfAbs.plot.bar(rot=0,color='green')
```

Out[229]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1931046df60>
```



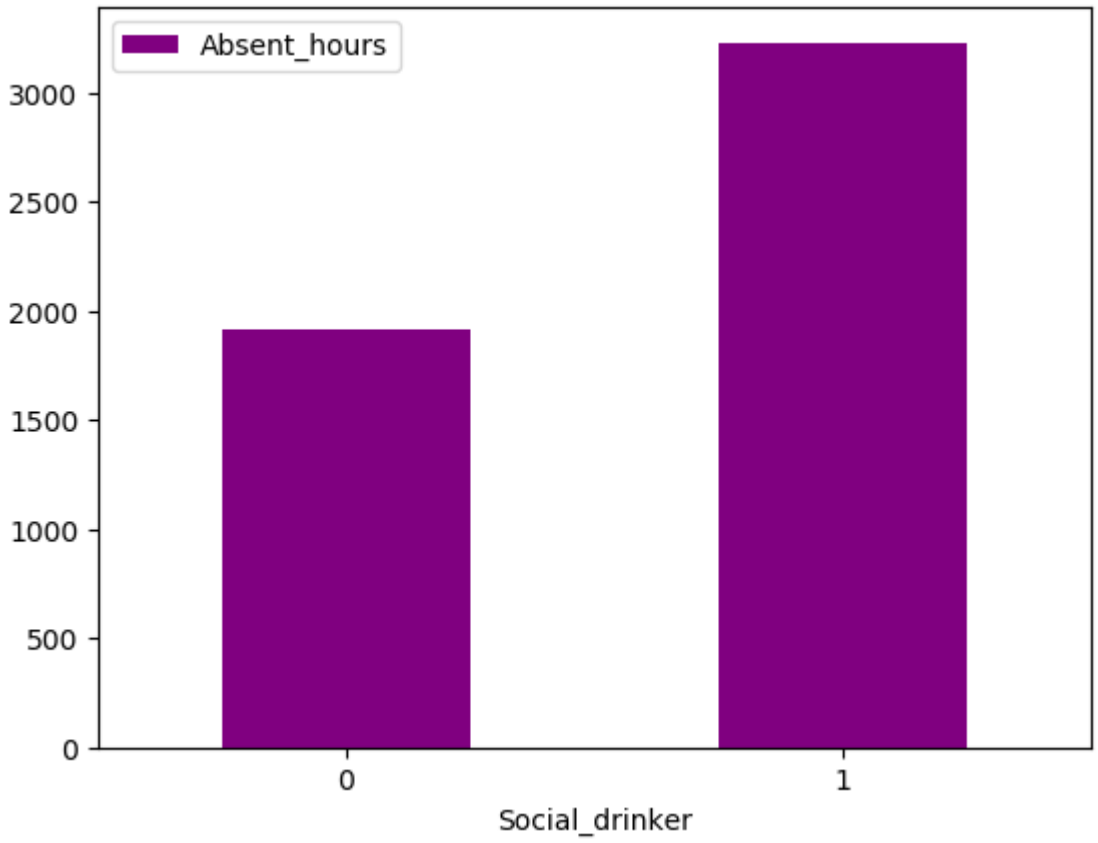
In [231]:
```
edfSon=edf.groupby('Son',as_index=True).sum()[['Absent_hours']]
edfSon
edfSon.plot.bar(rot=0,color='purple')
```

Out[231]: <matplotlib.axes._subplots.AxesSubplot at 0x1931062fc88>



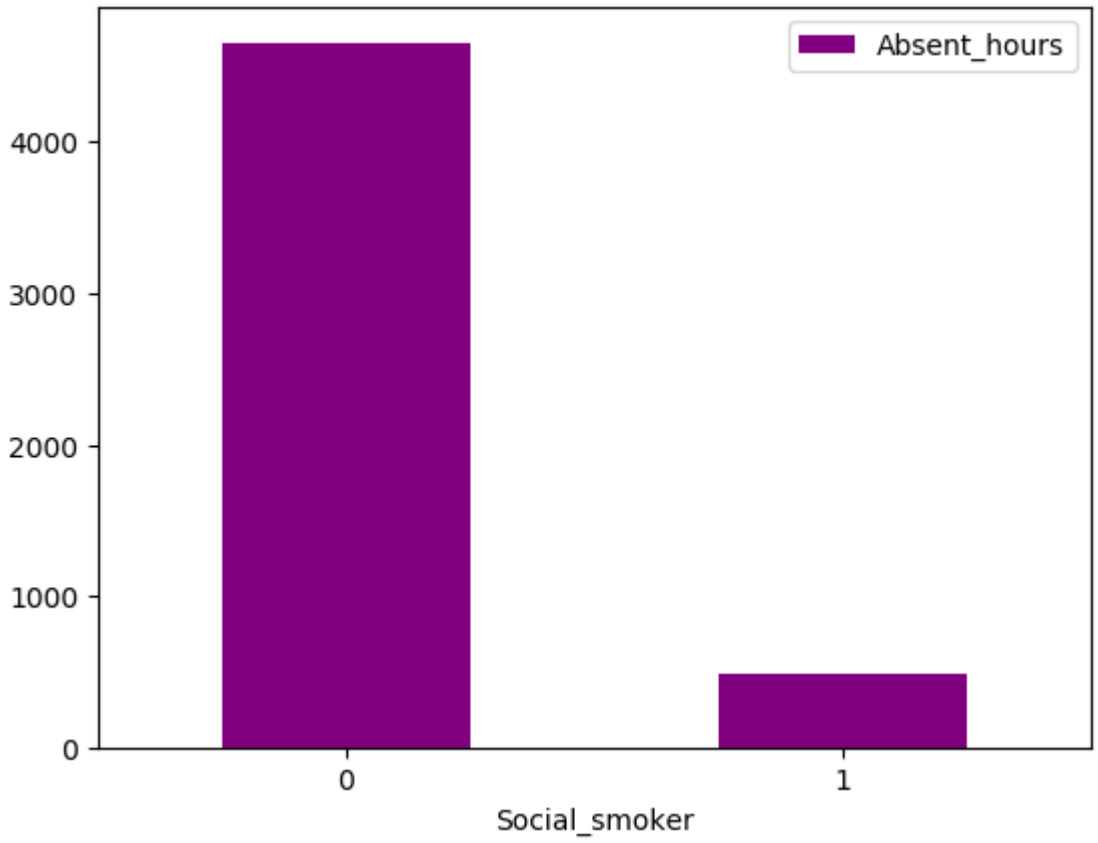
In [232]:
```
edfSD=edf.groupby('Social_drinker',as_index=True).sum()[['Absent_hours']]
edfSD
edfSD.plot.bar(rot=0,color='purple')
```

Out[232]: <matplotlib.axes._subplots.AxesSubplot at 0x19310682ba8>



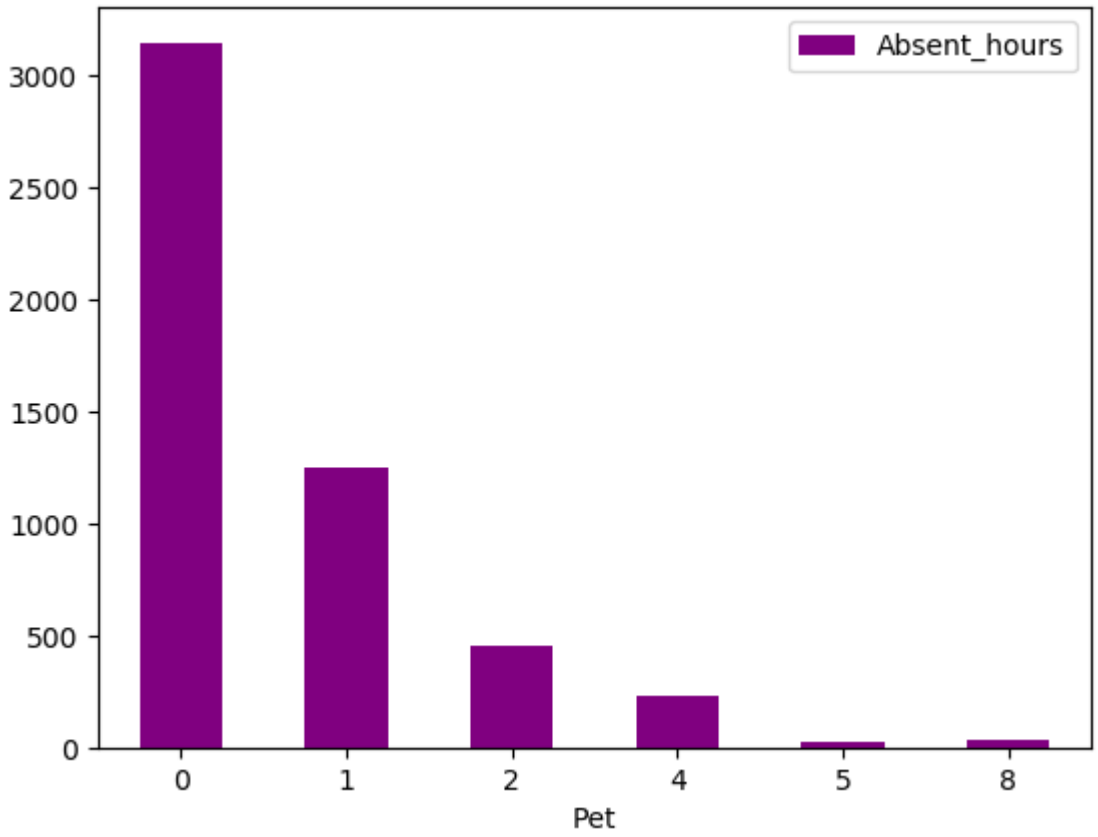
In [233]:
```
edfSS=edf.groupby('Social_smoker',as_index=True).sum()[['Absent_hours']]
edfSS
edfSS.plot.bar(rot=0,color='purple')
```

Out[233]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19311707160>
```



In [234]:
```
edfPet=edf.groupby('Pet',as_index=True).sum()[['Absent_hours']]
edfPet
edfPet.plot.bar(rot=0,color='purple')
```

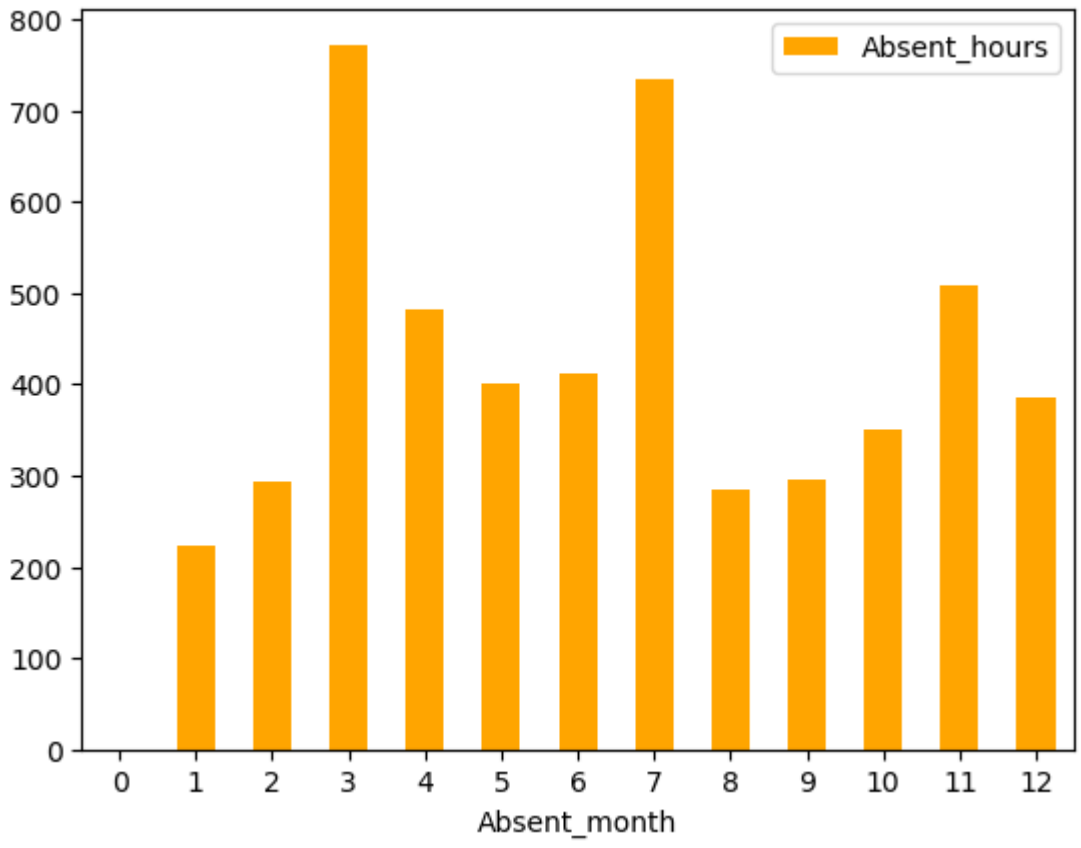
Out[234]: <matplotlib.axes._subplots.AxesSubplot at 0x1931175e2b0>



In [241]:
```
edfmonth=edf.groupby('Absent_month',as_index=True).sum()[['Absent_hours']]
edfmonth
edfmonth.plot.bar(rot=0,color='orange')
edfmonth
```

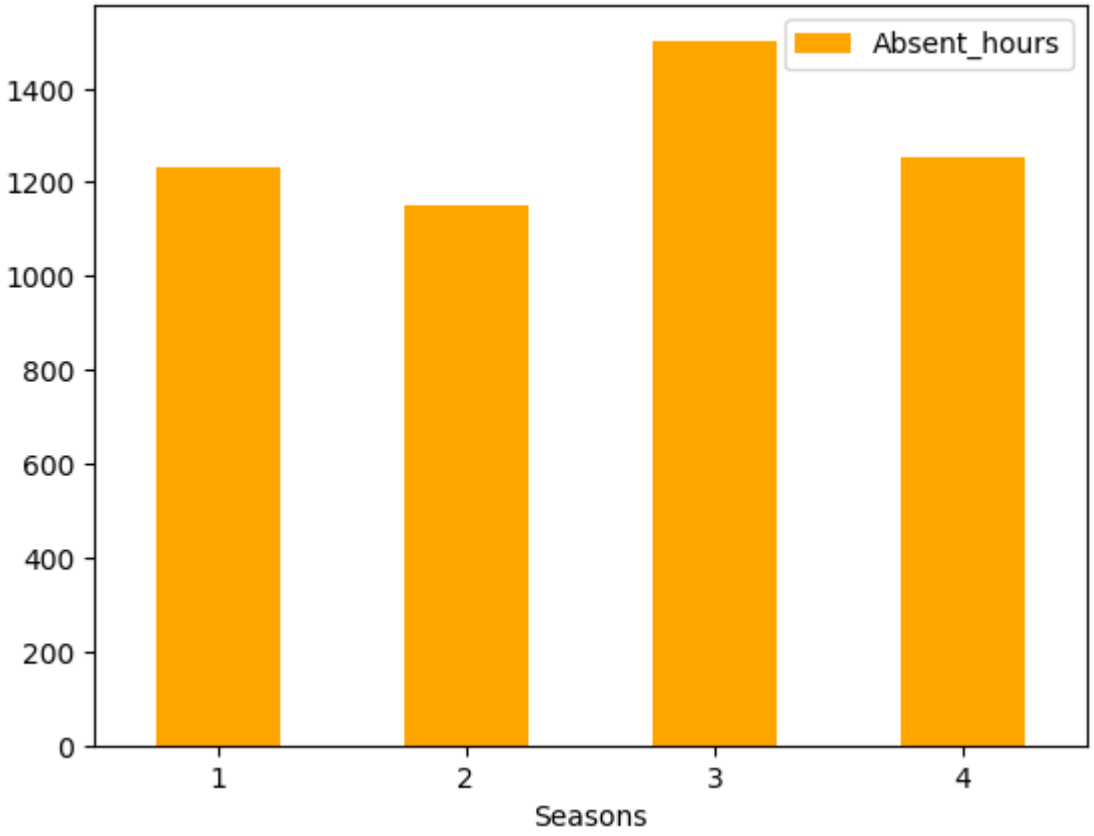
Out[241]:

Absent_month	Absent_hours
0	0
1	222
2	294
3	773
4	482
5	402
6	411
7	734
8	284
9	296
10	350
11	509
12	385



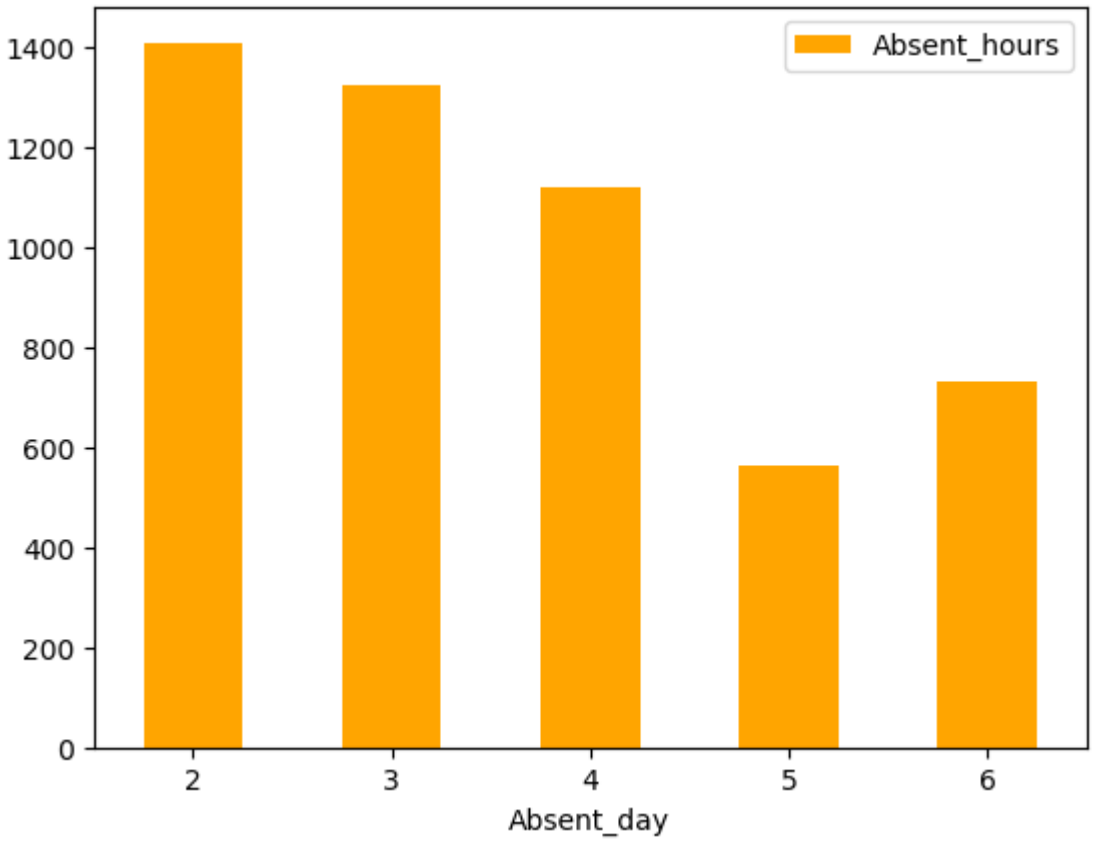
In [239]:
```
edfseason=edf.groupby('Seasons',as_index=True).sum()[['Absent_hours']]

edfseason
edfseason.plot.bar(rot=0,color='orange')
```

Out[239]: <matplotlib.axes._subplots.AxesSubplot at 0x193119ae710>



In [240]:
```
edfday=edf.groupby('Absent_day',as_index=True).sum()[['Absent_hours']]
edfday
edfday.plot.bar(rot=0,color='orange')
```

Out[240]: <matplotlib.axes._subplots.AxesSubplot at 0x19311a39780>



In [ ]: