DATA PREPROCESCING

Cleaning the Reason for Absence

Reason for Absence	Date	Transportation Expense	Distance to Work	Age	Daily W
26	07/07/2015	289	36	33	
0	14/07/2015	118	13	50	
23	15/07/2015	179	51	38	
7	16/07/2015	279	5	39	
23	23/07/2015	289	36	33	
23	10/07/2015	179	51	38	
22	17/07/2015	361	52	28	
23	24/07/2015	260	50	36	
19	06/07/2015	155	12	34	
22	13/07/2015	235	11	37	

	Reason_1	Reason_2	Reason_3	Reason_4	Date
0	0	0	0	1	07/07/2015
1	0	0	0	0	14/07/2015
2	0	0	0	1	15/07/2015
3	1	0	0	0	16/07/2015
4	0	0	0	1	23/07/2015

1. Create a separate dataframe, containing dummy values for ALL available reasons

```
reason_columns = pd.get_dummies(df['Reason for Absence'], drop_first = True)
```

2. Split reason_columns into 4 types

```
reason_type_1 = reason_columns.loc[:,1:14].max(axis=1)
reason_type_2 = reason_columns.loc[:,15:17].max(axis=1)
reason_type_3 = reason_columns.loc[:,18:21].max(axis=1)
reason_type_4 = reason_columns.loc[:,22:].max(axis=1)
```

3. To avoid multicollinearity, drop the 'Reason for Absence' column from df

```
df = df.drop(['Reason for Absence'], axis = 1)
```

4. Concatenate df and the 4 types of reason for absence

```
df = pd.concat([df, reason_type_1, reason_type_2, reason_type_3, reason_type_4], axis = 1)
```

5. Assign names to the 4 reason type columns

```
column_names = ['Date', 'Transportation Expense', 'Distance to Work', 'Age',
                'Daily Work Load Average', 'Body Mass Index', 'Education', 'Children',
                'Pet', 'Absenteeism Time in Hours', 'Reason_1', 'Reason_2', 'Reason_3', 'Reason_4']
df.columns = column_names
```

6. Re-order the columns in df

```
column_names_reordered = ['Reason_1', 'Reason_2', 'Reason_3', 'Reason_4', 'Date', 'Transportation
Expense',
                          'Distance to Work', 'Age', 'Daily Work Load Average', 'Body Mass Index', 'Education',
                          'Children', 'Pet', 'Absenteeism Time in Hours']
df = df[column_names_reordered]
```

DATA PREPROCESCING

Cleaning the Date

	Reason_1	Reason_2	Reason_3	Reason_4	Date	Transportation Expense	Distance to Work	Age
0	0	0	0	1	07/07/2015	289	36	33
1	0	0	0	0	14/07/2015	118	13	50
2	0	0	0	1	15/07/2015	179	51	38
3	1	0	0	0	16/07/2015	279	5	39
4	0	0	0	1	23/07/2015	289	36	33
5	0	0	0	1	10/07/2015	179	51	38
6	0	0	0	1	17/07/2015	361	52	28
7	0	0	0	1	24/07/2015	260	50	36
8	0	0	1	0	06/07/2015	155	12	34

	Reason_1	Reason_2	Reason_3	Reason_4	Month Value	Day of the Week	Transportation Expense	Distance to Work	Age
0	0	0	0	1	7	1	289	36	33
1	0	0	0	0	7	1	118	13	50
2	0	0	0	1	7	2	179	51	38
3	1	0	0	0	7	3	279	5	39
4	0	0	0	1	7	3	289	36	33
5	0	0	0	1	10	2	179	51	38
6	0	0	0	1	7	4	361	52	28
7	0	0	0	1	7	4	260	50	36
8	0	0	1	0	6	6	155	12	34

1. convert the 'Date' column into datetime

```
df['Date'] = pd.to_datetime(df['Date'], format='%d/%m/%Y')
```

2. create a list with month values retrieved from the 'Date' column

```
list_months = []
for i in range(df.shape[0]):
    list_months.append(df['Date'][i].month)
```

3. insert the values in a new column in df, called 'Month Value'

```
df['Month Value'] = list_months
```

4. create a new feature called 'Day of the Week'

```
df['Day of the Week'] = df['Date'].apply(lambda x: x.weekday())
```

5. drop the 'Date' column from df

```
df = df.drop(['Date'], axis = 1)
```

6. re-order the columns in df

```
column_names_upd = ['Reason_1', 'Reason_2', 'Reason_3', 'Reason_4', 'Month Value', 'Day of the Week',
                    'Transportation Expense', 'Distance to Work', 'Age',
                    'Daily Work Load Average', 'Body Mass Index', 'Education', 'Children',
                    'Pet', 'Absenteeism Time in Hours']
df = df[column_names_upd]
```