In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
data = pd.read_csv(r"C:\Users\Gokul\Downloads\kerala.csv")
data.tail()
```

Out[2]:

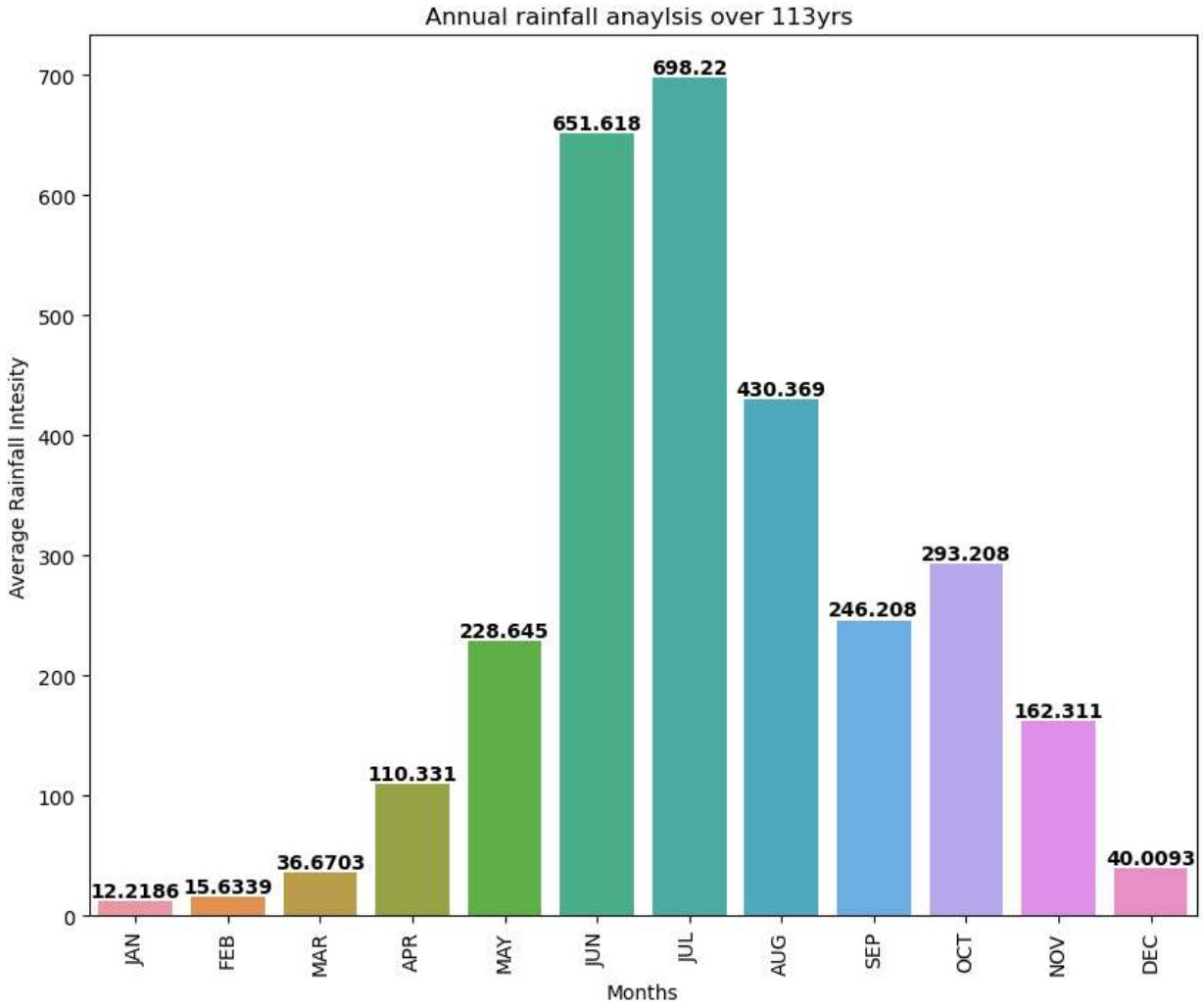
	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	D
113	KERALA	2014	4.6	10.3	17.9	95.7	251.0	454.4	677.8	733.9	298.8	355.5	99.5	4
114	KERALA	2015	3.1	5.8	50.1	214.1	201.8	563.6	406.0	252.2	292.9	308.1	223.6	7
115	KERALA	2016	2.4	3.8	35.9	143.0	186.4	522.2	412.3	325.5	173.2	225.9	125.4	2
116	KERALA	2017	1.9	6.8	8.9	43.6	173.5	498.5	319.6	531.8	209.5	192.4	92.5	3
117	KERALA	2018	29.1	52.1	48.6	116.4	183.8	625.4	1048.5	1398.9	423.6	356.1	125.4	6

In [3]:
```
data.columns = [c.replace(' ANNUAL RAINFALL', 'ANNUAL_RAINFALL') for c in data.columns]
data.head()
```

Out[3]:

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DE
0	KERALA	1901	28.7	44.7	51.6	160.0	174.7	824.6	743.0	357.5	197.7	266.9	350.8	48
1	KERALA	1902	6.7	2.6	57.3	83.9	134.5	390.9	1205.0	315.8	491.6	358.4	158.3	121
2	KERALA	1903	3.2	18.6	3.1	83.6	249.7	558.6	1022.5	420.2	341.8	354.1	157.0	59
3	KERALA	1904	23.7	3.0	32.2	71.5	235.7	1098.2	725.5	351.8	222.7	328.1	33.9	3
4	KERALA	1905	1.2	22.3	9.4	105.9	263.3	850.2	520.5	293.6	217.2	383.5	74.4	0

In [4]:
```
cols = ['JAN', 'FEB', 'MAR', 'APR','MAY','JUN','JUL','AUG','SEP','OCT','NOV','DEC']
monthly_avg = data[cols].mean()
monthly_avg
x=monthly_avg.index
y=monthly_avg
plt.figure(figsize=(10,8))
ax1 = sns.barplot(x = x ,y = y,width=0.8)
for i in ax1.containers:
    ax1.bar_label(i, label_type='edge', fontsize=10, color='black', weight='bold')
plt.xticks(rotation=90,fontsize=10)
plt.xlabel("Months")
plt.ylabel("Average Rainfall Intesity")
plt.title("Annual rainfall anaylsis over 113yrs ")
plt.show()
```



In [5]:
```
# Here, the required columns(year and all 12 months) has been selected from the whole
columns = data.columns.tolist()
data2 = data[columns[1:14]]
data2.head()
```

Out[5]:

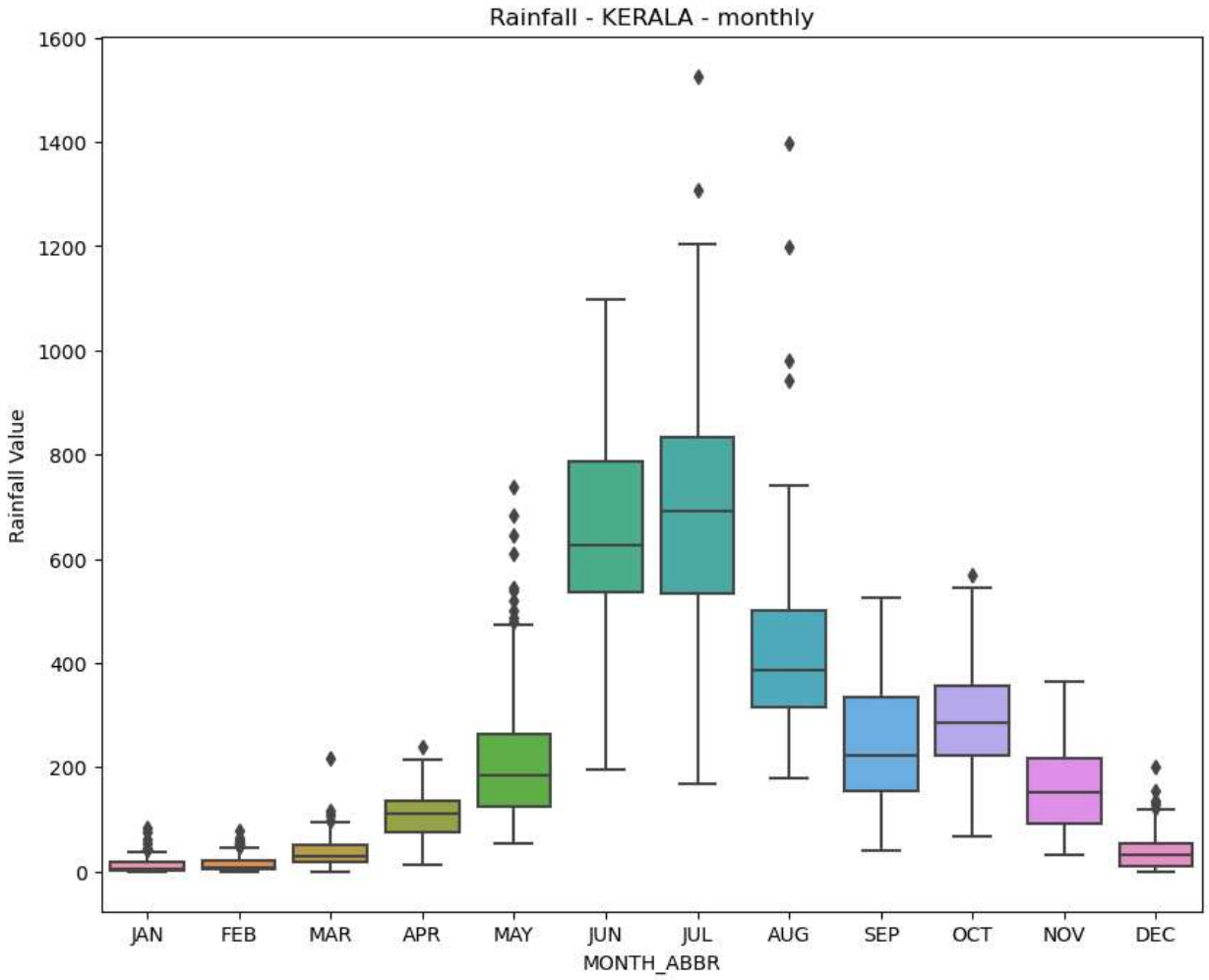
	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
0	1901	28.7	44.7	51.6	160.0	174.7	824.6	743.0	357.5	197.7	266.9	350.8	48.4
1	1902	6.7	2.6	57.3	83.9	134.5	390.9	1205.0	315.8	491.6	358.4	158.3	121.5
2	1903	3.2	18.6	3.1	83.6	249.7	558.6	1022.5	420.2	341.8	354.1	157.0	59.0
3	1904	23.7	3.0	32.2	71.5	235.7	1098.2	725.5	351.8	222.7	328.1	33.9	3.3
4	1905	1.2	22.3	9.4	105.9	263.3	850.2	520.5	293.6	217.2	383.5	74.4	0.2

In [9]:
```
# In order to have a better undersatnding aboiut the data, the curent data is pivoted
pivot_data  = pd.melt(data2,
                id_vars = ['YEAR'],
                value_vars = ['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL','AUG', 'S
               var_name='MONTH_ABBR', value_name='VALUE')
```

In [ ]:
```
fig, ax = plt.subplots(1, 1, figsize=(10, 8))
sns.boxplot(data=pivot_data, x='MONTH_ABBR', y= pivot_data.VALUE, ax=ax)
```

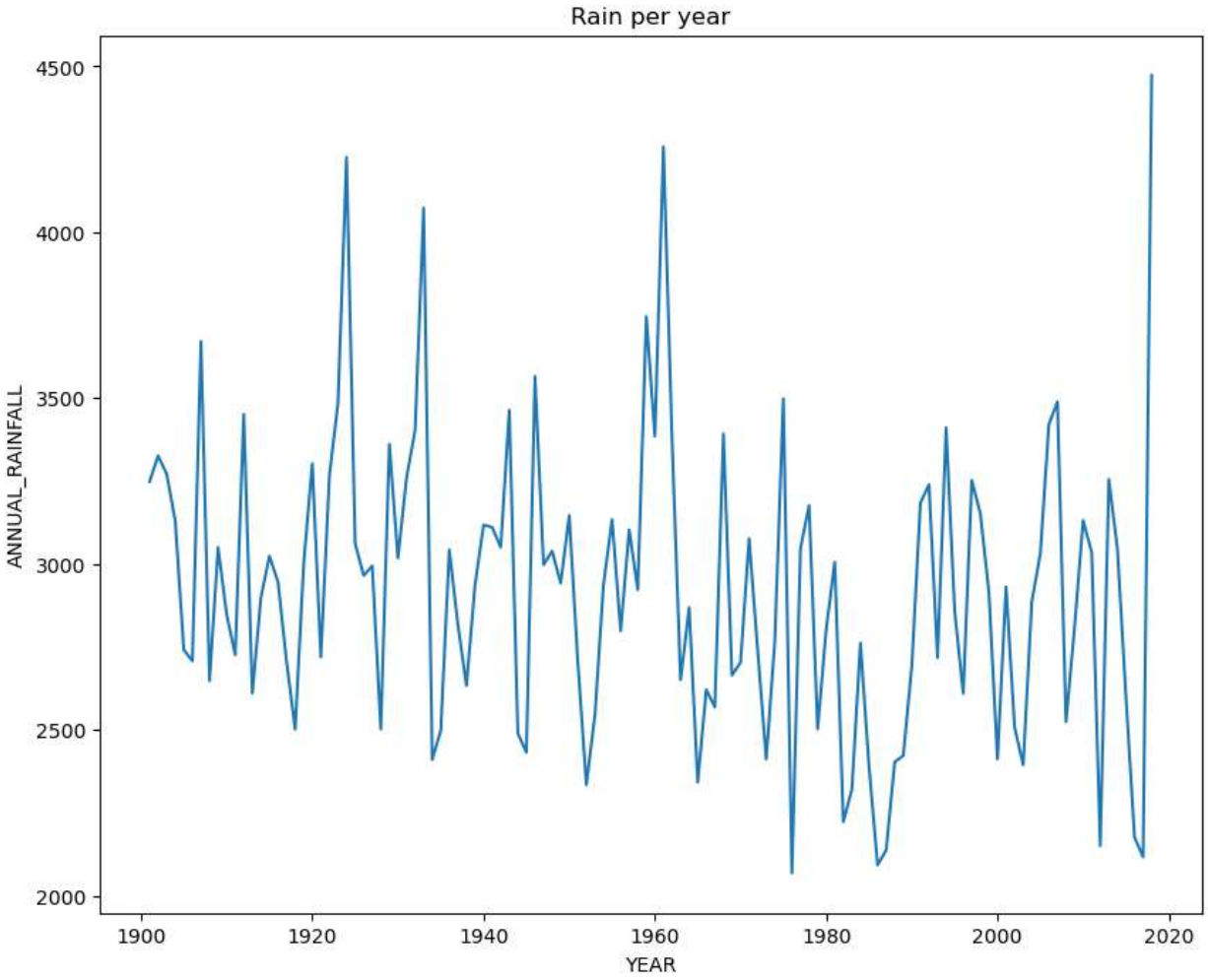
```
ax.set_ylabel('Rainfall Value')
ax.set_title('Rainfall - KERALA - monthly')
```

Out[ ]:
```
Text(0.5, 1.0, 'Rainfall - KERALA - monthly')
```



In [ ]:
```
plt.figure(figsize=(10,8))
sns.lineplot(data=data,
             x="YEAR",
             y="ANNUAL_RAINFALL")
plt.title("Rain per year")
```

Out[ ]:
```
Text(0.5, 1.0, 'Rain per year')
```



In [7]:
```
# Based on the graph, it is understood that in an year most rain impacted months are j

impactful_columns = ['YEAR', 'JUN', 'JUL', 'OCT', 'ANNUAL_RAINFALL', 'FLOODS']

impactful_columns
```

Out[7]:
```
['YEAR', 'JUN', 'JUL', 'OCT', 'ANNUAL_RAINFALL', 'FLOODS']
```

In [10]:
```
mdata = data[impactful_columns]

mdata.head()
```

Out[10]:

	YEAR	JUN	JUL	OCT	ANNUAL_RAINFALL	FLOODS
0	1901	824.6	743.0	266.9	3248.6	YES
1	1902	390.9	1205.0	358.4	3326.6	YES
2	1903	558.6	1022.5	354.1	3271.2	YES
3	1904	1098.2	725.5	328.1	3129.7	YES
4	1905	850.2	520.5	383.5	2741.6	NO

In [14]:
```
threshold_jun = mdata['JUN'].median()
threshold_jul = mdata['JUL'].median()
threshold_oct = mdata['OCT'].median()
```

```
threshold_ar = mdata['ANNUAL_RAINFALL'].median()
round(threshold_jun,2), threshold_jul, threshold_oct, threshold_ar
```

Out[14]: (625.6, 691.65, 284.3, 2934.3)

In [16]:
```
thresholds = {
    'JUN': 625,
    'JUL': 691,
    'OCT': 284,
    'ANNUAL_RAINFALL': 2934
}

# Convert columns to binary based on thresholds
for col, threshold in thresholds.items():
    mdata[col] = (data[col] > threshold).astype(int)

mdata.head()
```

```
C:\Users\Gokul\AppData\Local\Temp\ipykernel_5360\3535735778.py:10: SettingWithCopyWar
ning:
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/us
er_guide/indexing.html#returning-a-view-versus-a-copy
  mdata[col] = (data[col] > threshold).astype(int)
C:\Users\Gokul\AppData\Local\Temp\ipykernel_5360\3535735778.py:10: SettingWithCopyWar
ning:
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/us
er_guide/indexing.html#returning-a-view-versus-a-copy
  mdata[col] = (data[col] > threshold).astype(int)
C:\Users\Gokul\AppData\Local\Temp\ipykernel_5360\3535735778.py:10: SettingWithCopyWar
ning:
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/us
er_guide/indexing.html#returning-a-view-versus-a-copy
  mdata[col] = (data[col] > threshold).astype(int)
C:\Users\Gokul\AppData\Local\Temp\ipykernel_5360\3535735778.py:10: SettingWithCopyWar
ning:
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/us
er_guide/indexing.html#returning-a-view-versus-a-copy
  mdata[col] = (data[col] > threshold).astype(int)
```

Out[16]:

	YEAR	JUN	JUL	OCT	ANNUAL_RAINFALL	FLOODS
0	1901	1	1	0	1	YES
1	1902	0	1	1	1	YES
2	1903	0	1	1	1	YES
3	1904	1	1	1	1	YES
4	1905	1	0	1	0	NO

In [17]:
```
mask = ((mdata['JUN']==1) | (mdata['JUL']==1) | (mdata['OCT']==1) & (mdata['ANNUAL_RAI
mask
```

Out[17]:
```
0       True
1       True
2       True
3       True
4       True
       ...
113     True
114    False
115    False
116    False
117     True
Length: 118, dtype: bool
```

In [18]:
```
mdata = mdata[mask]

mdata.head()
```

Out[18]:

	YEAR	JUN	JUL	OCT	ANNUAL_RAINFALL	FLOODS
0	1901	1	1	0	1	YES
1	1902	0	1	1	1	YES
2	1903	0	1	1	1	YES
3	1904	1	1	1	1	YES
4	1905	1	0	1	0	NO

In [19]:
```
pd.crosstab(index = mdata['JUN'],
            columns = mdata['FLOODS'],
            margins=True,
            margins_name='Total')
```

Out[19]:

FLOODS	NO	YES	Total
JUN			
0	17	13	30
1	16	44	60
Total	33	57	90

In [20]:
```
pd.crosstab(index = mdata['JUL'],
            columns = mdata['FLOODS'],
            margins=True,
            margins_name='Total')
```

Out[20]:

FLOODS	NO	YES	Total
JUL			
0	13	18	31
1	20	39	59
Total	33	57	90

In [21]:
```
pd.crosstab(index = [mdata['JUL'],mdata['JUN']],
            columns = mdata['FLOODS'],
            margins=True,
            margins_name='Total')
```

Out[21]:

	FLOODS	NO	YES	Total
JUL	JUN			
0	0	0	1	1
	1	13	17	30
1	0	17	12	29
	1	3	27	30
Total		33	57	90