In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,mean_squared_error
from sklearn.model_selection import train_test_split
import seaborn as sns
```

In [2]:

```
source = 'http://bit.ly/w-data'
dataset=pd.read_csv(source)
```

In [3]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   25 non-null     float64
 1   Scores  25 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 528.0 bytes
```

In [4]:

```
dataset.describe()
```

Out[4]:

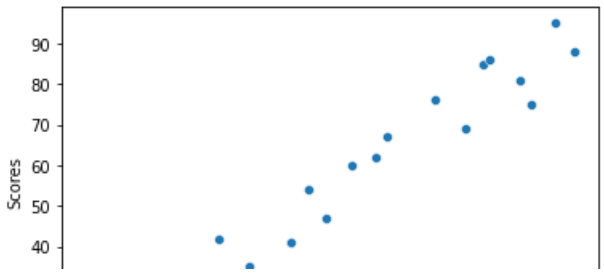
	Hours	Scores
count	25.000000	25.000000
mean	5.012000	51.480000
std	2.525094	25.286887
min	1.100000	17.000000
25%	2.700000	30.000000
50%	4.800000	47.000000
75%	7.400000	75.000000
max	9.200000	95.000000

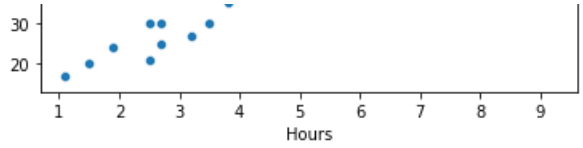
In [5]:

```
sns.scatterplot(x='Hours',y='Scores',data=dataset)
```

Out[5]:

```
<AxesSubplot:xlabel='Hours', ylabel='Scores'>
```



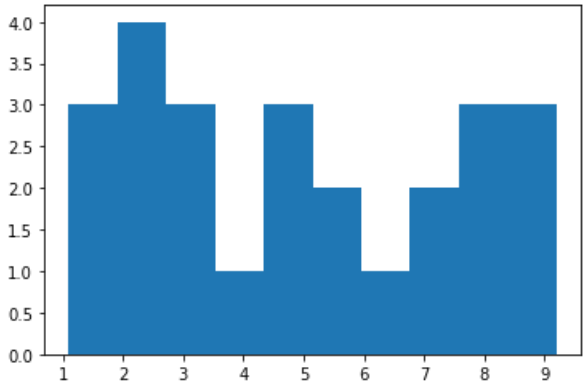


In [6]:

```
plt.hist(dataset['Hours'])
```

Out[6]:

```
(array([3., 4., 3., 1., 3., 2., 1., 2., 3., 3.]),
 array([1.1 , 1.91, 2.72, 3.53, 4.34, 5.15, 5.96, 6.77, 7.58, 8.39, 9.2 ]),
 <BarContainer object of 10 artists>)
```

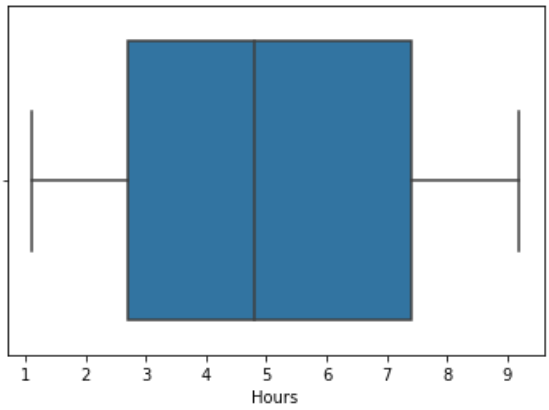


In [7]:

```
sns.boxplot(x='Hours',data=dataset)
```

Out[7]:

```
<AxesSubplot:xlabel='Hours'>
```

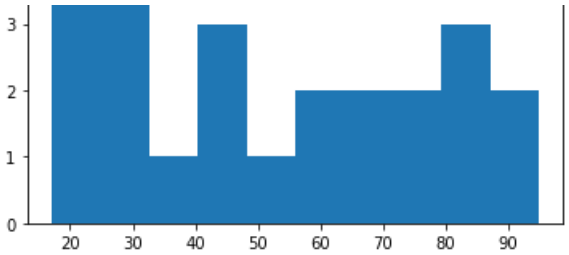


In [8]:

```
plt.hist(dataset['Scores'])
```

Out[8]:

```
(array([4., 5., 1., 3., 1., 2., 2., 2., 3., 2.]),
 array([17. , 24.8, 32.6, 40.4, 48.2, 56. , 63.8, 71.6, 79.4, 87.2, 95. ]),
 <BarContainer object of 10 artists>)
```

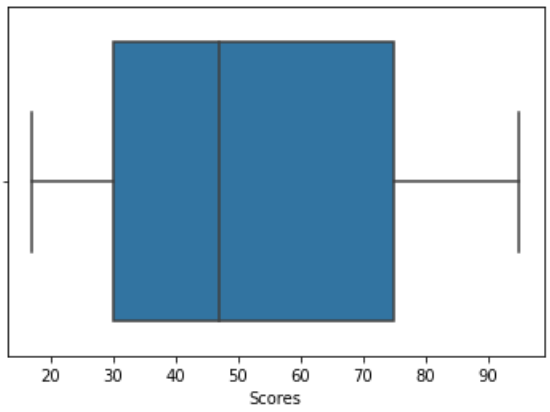


In [9]:

```
sns.boxplot(x='Scores',data=dataset)
```

Out[9]:

```
<AxesSubplot:xlabel='Scores'>
```



In [10]:

```
y=dataset['Scores'].copy()
X=dataset.drop('Scores',axis=1)
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=5)
```

In [11]:

```
lr=LinearRegression()
lr.fit(X_train,y_train)
```

Out[11]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [12]:

```
y_pred=lr.predict(X_test)
```

In [13]:

```
r2_score(y_test,y_pred)
```

Out[13]:

```
0.9248556597026296
```

In [14]:

```
lm_mse=mean_squared_error(y_test,y_pred)
lm_rmse=np.sqrt(lm_mse)
lm_rmse
```

Out[14]:

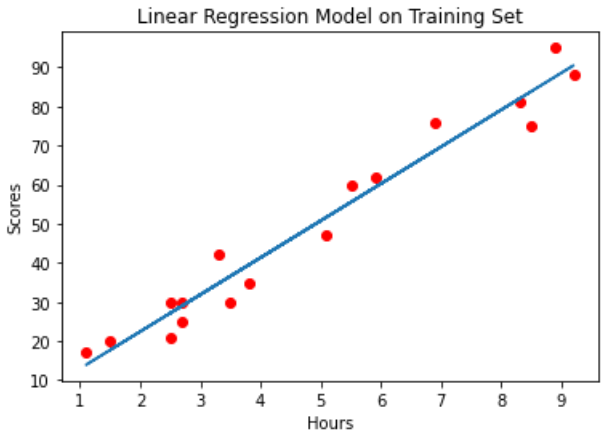
6.22229436847076

In [15]:

```
plt.scatter(X_train,y_train,color='red')
plt.plot(X_train,lr.predict(X_train))
plt.title('Linear Regression Model on Training Set')
plt.xlabel('Hours')
plt.ylabel('Scores')
```

Out[15]:

```
Text(0, 0.5, 'Scores')
```

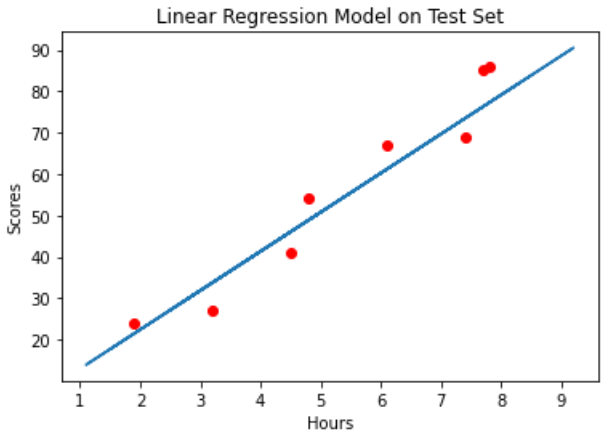


In [16]:

```
plt.scatter(X_test,y_test,color='red')
plt.plot(X_train,lr.predict(X_train))
plt.title('Linear Regression Model on Test Set')
plt.xlabel('Hours')
plt.ylabel('Scores')
```

Out[16]:

```
Text(0, 0.5, 'Scores')
```



In [17]:

```
lr.predict([[9.25]])
```

Out[17]:

```
array([90.96001897])
```

In [18]:

```
lr.coef_
```

Out[18]:

```
array([9.45348802])
```

In [19]:

```
lr.intercept_
```

Out[19]:

```
3.5152547646049186
```

In [ ]:

```

```