EXPLORATORY DATA ANALYSIS ON STUDENT DATASET

WHAT IS EDA:

```
 Exploratory Data Analysis is a data analytic process that aims to understand the data in
depth and learn its different characterstics.
```

```
STEPS INVOLVED IN EXPLORATORY DATA ANALYSIS:
1.Understand the data :  understanding the variables in the dataset,and on what kind of data
you are working with.
2.Clean the data      : cleaning data from redundancy,irregularity and deleting unnecessary
columns,outliers which causes noise in the data.
3.Analysis of Relationship between variables : analysing the relationship between the
variables in the dataset.
4.Data visualisation  : visualizing the relationship in different patterns to understand
easily.
```

STUDENT DATASET:

```
     This dataset tells about the students and this data is taken from kaggle website and we
will try to understand
     the dataset by using pandas,numpy for data storing and processing  and for visualisation
we use matplotlib and seaborn.
```

The data contains 6 coulmns:

1.Hours studied : no of hours studied by student.

2.previous scores: The previous scores of student.

3.extracircular activites: the student has extracircular activities or not.

4.sleep hours: the no.of hours they are sleeping.

5.sample question papers practiced: the no.of pervious questions papers they practiced.

6.performance index : the performance of the student.

In [ ]:
```
import  pandas as pd
import numpy as np
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
import matplotlib.pyplot as plt
%matplotlib inline
```

Here we are Reading the studentperformance.csv file with the help of pandas

In [12]:
```
studentperformance_df=pd.read_csv('Student_Performance.csv')
```

understanding the data

we use the head() command to retrieve the first 5 rows of our studentperformance data

In [13]:
```
studentperformance_df.head()
```

Out[13]:

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	Yes	9	1	91.0
1	4	82	No	4	2	65.0
2	8	51	Yes	7	2	45.0
3	5	52	Yes	5	2	36.0
4	7	75	No	8	5	66.0

we use the tail() command to retrieve the last 5 rows of our data

In [14]:
```
studentperformance_df.tail()
```

Out[14]:

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
9995	1	49	Yes	4	2	23.0
9996	7	64	Yes	8	5	58.0
9997	6	83	Yes	8	5	74.0
9998	9	97	Yes	7	0	95.0
9999	7	74	No	8	1	64.0

This info() command is used to retrieve the information i.e., whether the data has null values or not and datatype of eacch columns

In [15]:
```
studentperformance_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 6 columns):
 #   Column                            Non-Null Count  Dtype  
---  ------                            --------------  -----  
 0   Hours Studied                     10000 non-null  int64  
 1   Previous Scores                   10000 non-null  int64  
 2   Extracurricular Activities        10000 non-null  object 
 3   Sleep Hours                       10000 non-null  int64  
 4   Sample Question Papers Practiced  10000 non-null  int64  
 5   Performance Index                 10000 non-null  float64
dtypes: float64(1), int64(4), object(1)
memory usage: 468.9+ KB
```

This describe() command is used to display statistics values of our data(i.e.,mean,standard deviation,min,max) but this doesnot shows the statistics of objects in our data

In [16]:
```
studentperformance_df.describe()
```

Out[16]:

	Hours Studied	Previous Scores	Sleep Hours	Sample Question Papers Practiced	Performance Index
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	4.992900	69.445700	6.530600	4.583300	55.224800
std	2.589309	17.343152	1.695863	2.867348	19.212558
min	1.000000	40.000000	4.000000	0.000000	10.000000
25%	3.000000	54.000000	5.000000	2.000000	40.000000
50%	5.000000	69.000000	7.000000	5.000000	55.000000
75%	7.000000	85.000000	8.000000	7.000000	71.000000
max	9.000000	99.000000	9.000000	9.000000	100.000000

This describe(include='object') command used to display the count,unique values,top ,freq of each object in our data

In [17]:
```
studentperformance_df.describe(include='object')
```

Out[17]:

	Extracurricular Activities
count	10000
unique	2
top	No
freq	5052

This shape command is used to display the number of rows and columns of our data

In [18]:
```
studentperformance_df.shape
```

Out[18]:
```
(10000, 6)
```

columns command shows the column names

In [19]:
```
studentperformance_df.columns
```

Out[19]:
```
Index(['Hours Studied', 'Previous Scores', 'Extracurricular Activities',
       'Sleep Hours', 'Sample Question Papers Practiced', 'Performance Index'],
      dtype='object')
```

nunique() command displays the unique values in each column.

In [20]:
```
studentperformance_df.nunique()
```

Out[20]:
```
Hours Studied                      9
Previous Scores                   60
Extracurricular Activities         2
Sleep Hours                        6
Sample Question Papers Practiced  10
Performance Index                 91
dtype: int64
```

It defines the unique values of particular column.

In [21]:
```
studentperformance_df['Extracurricular Activities'].unique()
```

Out[21]:
```
array(['Yes', 'No'], dtype=object)
```

cleaning the data

By using isnull() we can identify null values and by using sum() we can find sum.

In [22]:
```
studentperformance_df.isnull().sum()
```

Out[22]:
```
Hours Studied                     0
Previous Scores                   0
Extracurricular Activities        0
Sleep Hours                       0
Sample Question Papers Practiced  0
Performance Index                 0
dtype: int64
```

we are defining a variable num_df which has date related to year,month,day,day_of_week to find correlation

In [23]:
```
num_df=studentperformance_df[['Hours Studied', 'Previous Scores',
       'Sleep Hours', 'Sample Question Papers Practiced','Performance Index']]
num_df
```

Out[23]:

	Hours Studied	Previous Scores	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	9	1	91.0
1	4	82	4	2	65.0
2	8	51	7	2	45.0
3	5	52	5	2	36.0
4	7	75	8	5	66.0
...	...	...	...	...	...
9995	1	49	4	2	23.0
9996	7	64	8	5	58.0
9997	6	83	8	5	74.0
9998	9	97	7	0	95.0
9999	7	74	8	1	64.0

10000 rows × 5 columns

corelation is used to find the relationship between two columns. we have created a variable corelation to find correlation of num_df by corr() command

In [24]:
```
corelation=num_df.corr()
corelation
```

Out[24]:

	Hours Studied	Previous Scores	Sleep Hours	Sample Question Papers Practiced	Performance Index
Hours Studied	1.000000	-0.012390	0.001245	0.017463	0.373730
Previous Scores	-0.012390	1.000000	0.005944	0.007888	0.915189
Sleep Hours	0.001245	0.005944	1.000000	0.003990	0.048106
Sample Question Papers Practiced	0.017463	0.007888	0.003990	1.000000	0.043268
Performance Index	0.373730	0.915189	0.048106	0.043268	1.000000

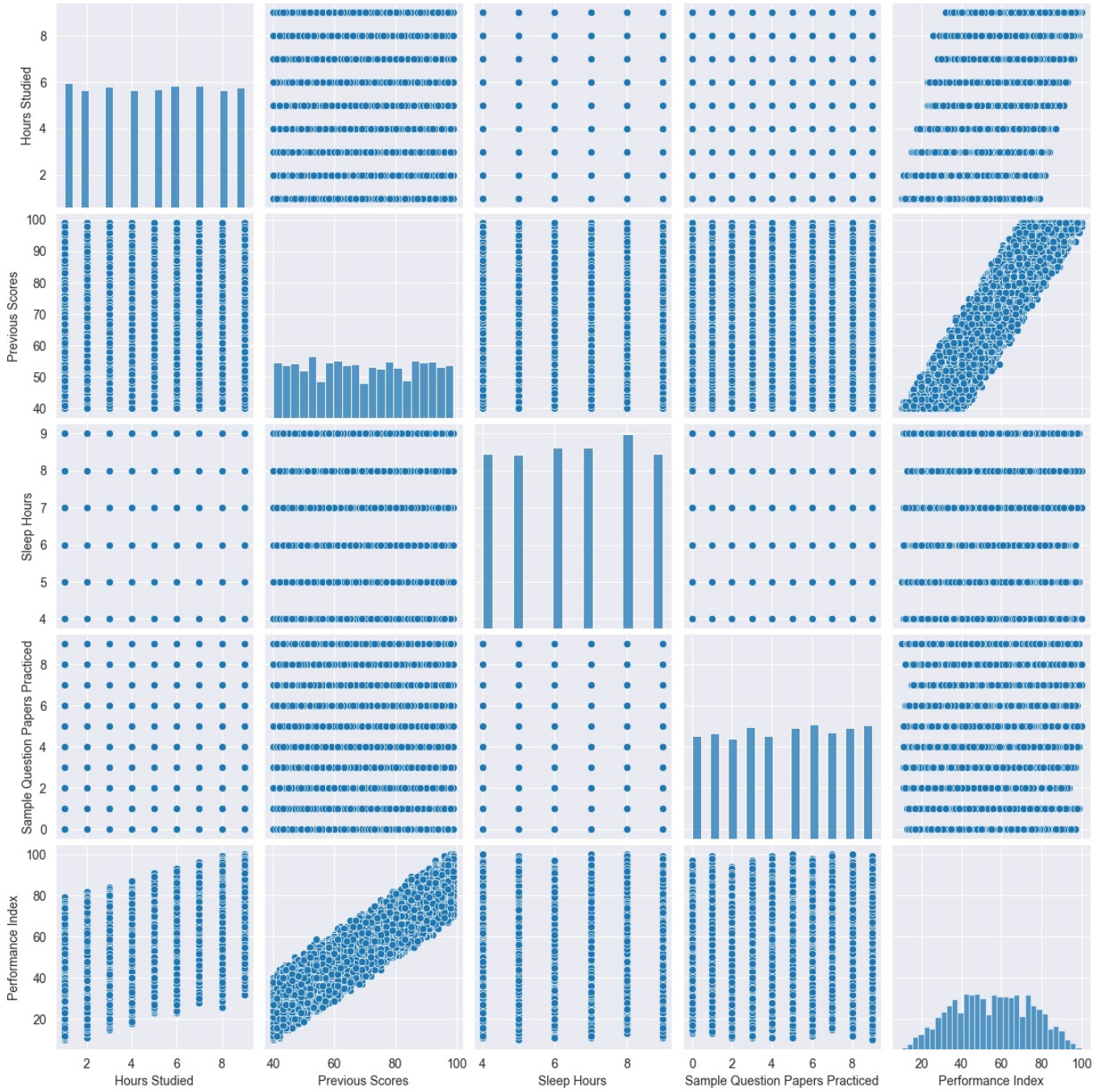
sns is a shorthand of seaborn.Here we are plotting heatmap.Heatmap is defined as a graphical representation of data using colors to visualize the value of the matrix.

In [69]:
```
plt.figure(figsize=(7,4))
sns.heatmap(corelation,xticklabels=corelation.columns,yticklabels=corelation.columns,
            annot=True,cmap='Blues')
plt.show()
```

Pairwaise plot

It will display a matrix of scatterplots for each pair of features in the dataframe,along with histograms,showing the relationships and distributions of the data.

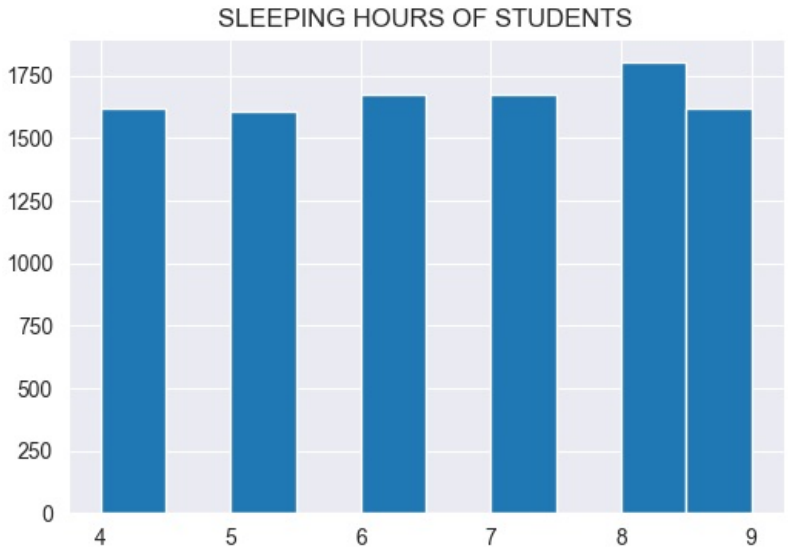
In [38]:
```
sns.pairplot(num_df);
plt.show()
```



Data visualisation.

It will display sleep hours to each student

In [41]:
```
plt.figure(figsize=(6,4))
plt.hist(x=studentperformance_df['Sleep Hours'],bins=10);
sns.set_style("darkgrid")
plt.title("SLEEPING HOURS OF STUDENTS");
plt.show()
```

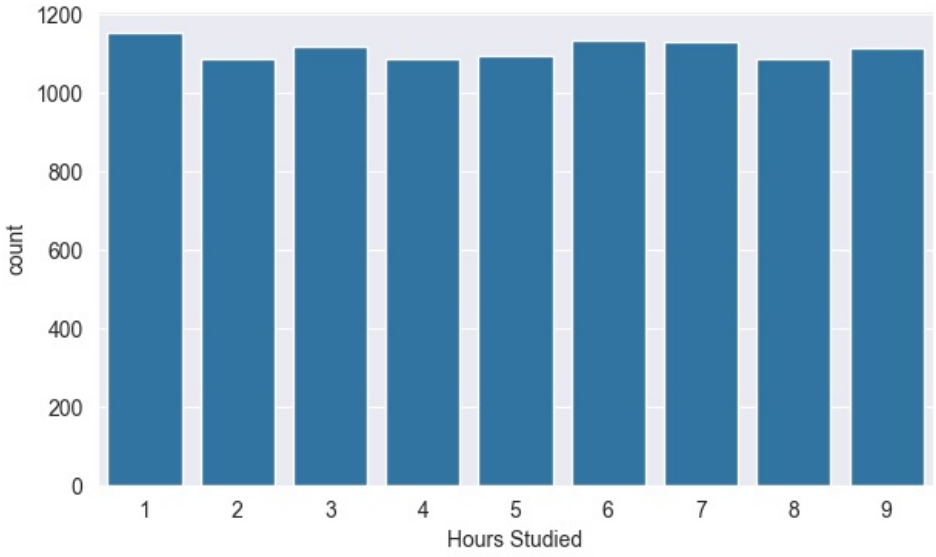


It shows the countplot of Hours Studied column.

In [46]:
```
plt.figure(figsize=(7,4))

sns.countplot(x='Hours Studied', data=studentperformance_df)
plt.show()
```

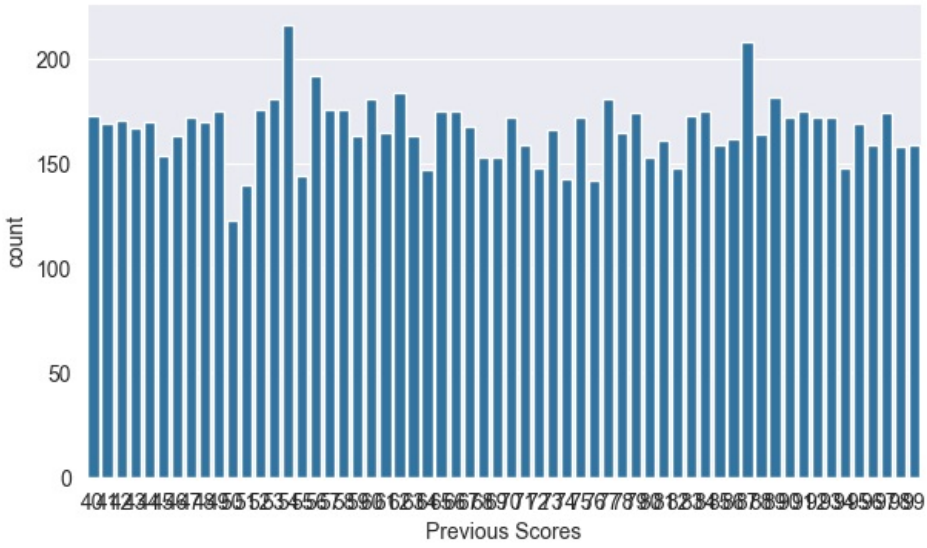
```
<Figure size 700x400 with 0 Axes>
<Figure size 700x400 with 0 Axes>
```



It shows the countplot of Previous Scores column.

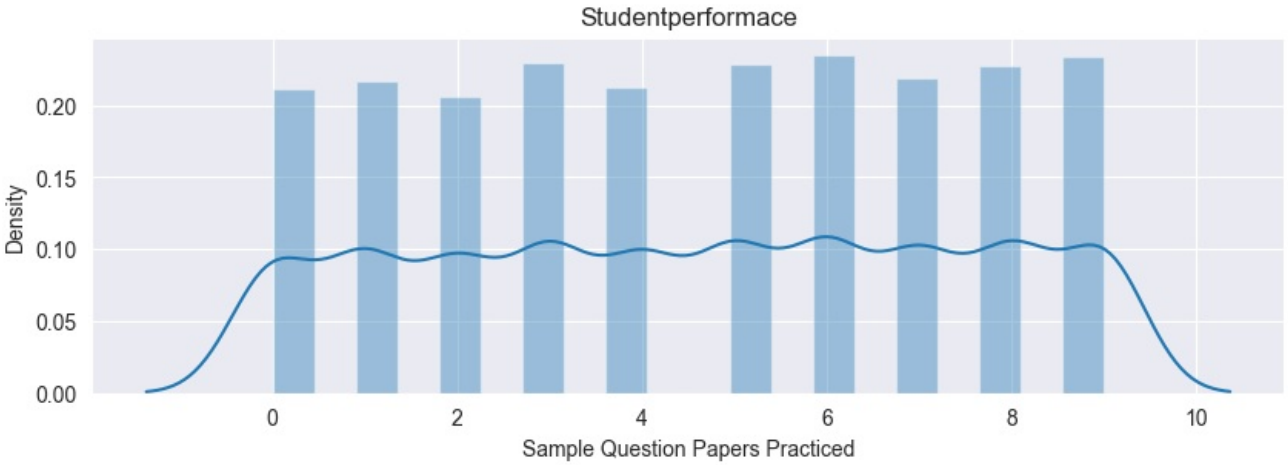
In [47]:
```
plt.figure(figsize=(7,4))

sns.countplot(x='Previous Scores', data=studentperformance_df)
plt.show()
```



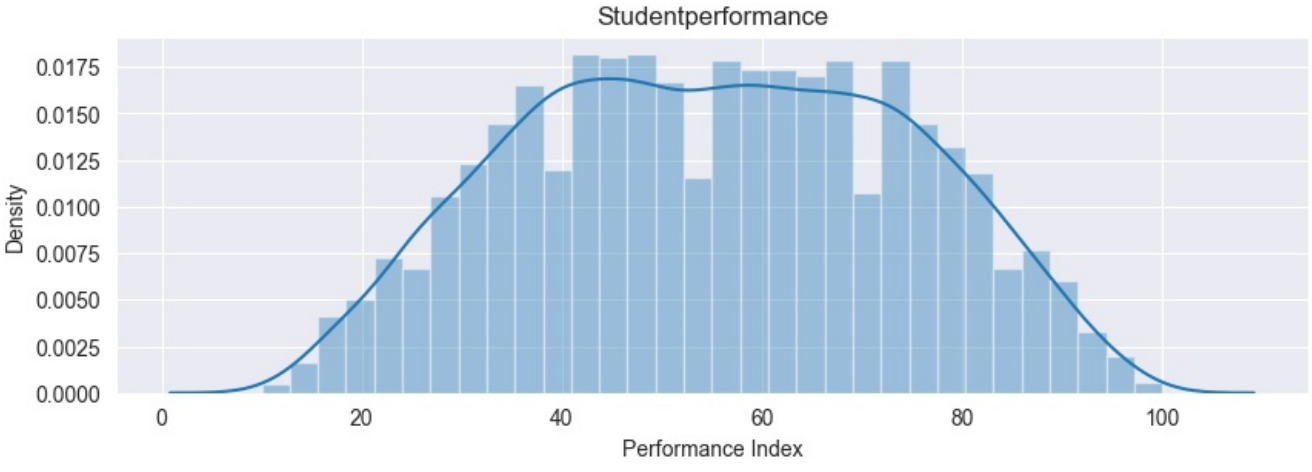
The above plot shows the density of sample question papers practiced.

In [48]:
```
plt.figure(figsize=(10,3))
plt.title("Studentperformace")
sns.distplot(studentperformance_df['Sample Question Papers Practiced'])
plt.show()
```



The above plot shows the density of perfoemance index practiced.

In [49]:
```
plt.figure(figsize=(10,3))
plt.title("Studentperformance")
sns.distplot(studentperformance_df['Performance Index'])
plt.show()
```



In [50]:
```
top_rated=studentperformance_df.nlargest(10,'Sleep Hours')
top_rated
```

Out[50]:

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	Yes	9	1	91.0
5	3	78	No	9	6	61.0
12	3	47	No	9	2	27.0
21	6	96	No	9	0	85.0
31	7	44	Yes	9	1	36.0
34	7	67	Yes	9	3	60.0
35	2	97	Yes	9	4	74.0
43	7	46	No	9	5	36.0
52	6	81	No	9	9	75.0
53	6	62	Yes	9	0	52.0

In [52]:
```
studentperformance_df['Sleep Hours'].unique()
```

Out[52]:
```
array([9, 4, 7, 5, 8, 6])
```

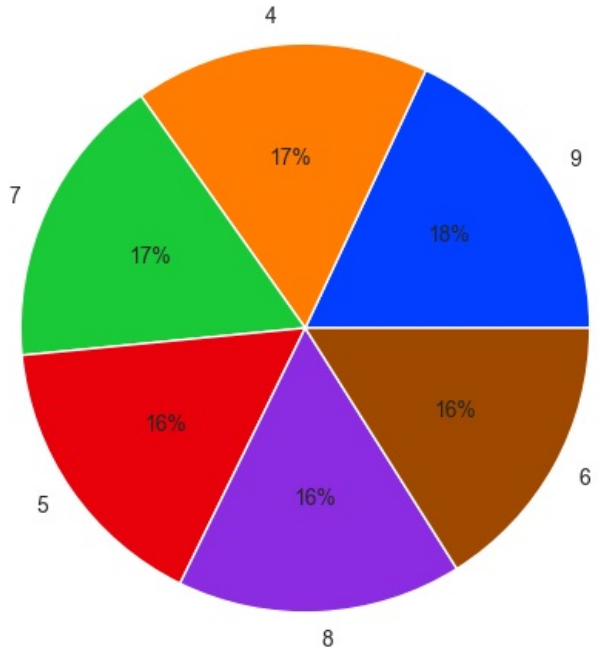
The above pie chart shows the how much percentage have sleep hours in dataframe.

In [65]:
```
plt.figure(figsize=(6,6))
data = studentperformance_df["Sleep Hours"].value_counts()

keys = [9, 4, 7, 5, 8, 6]

palette_color = sns.color_palette('bright')
plt.pie(data, labels=keys, colors=palette_color, autopct= '%.0f%%')
plt.show();
```

```
<Figure size 600x600 with 0 Axes>
<Figure size 600x600 with 0 Axes>
```



In [67]:
```
studentperformance_df['Extracurricular Activities'].unique()
```

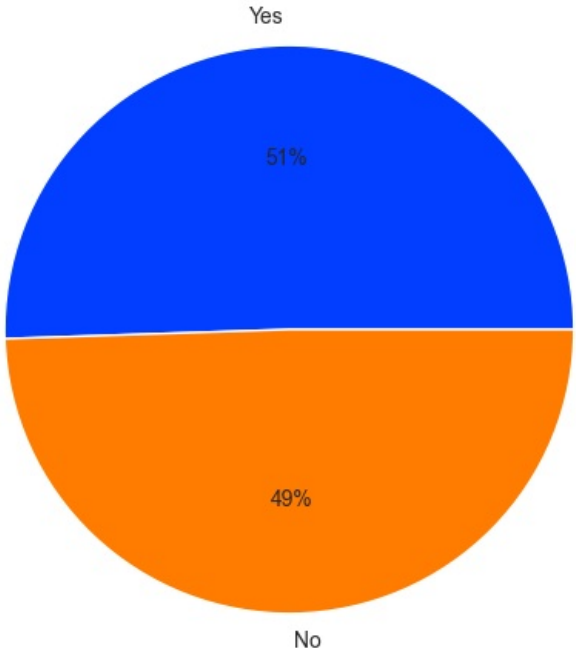
Out[67]:
```
array(['Yes', 'No'], dtype=object)
```

The above pie chart have Extracurricular Activities in dataframe.

In [68]:
```
plt.figure(figsize=(6,6))
data = studentperformance_df["Extracurricular Activities"].value_counts()

keys = ['Yes', 'No']

palette_color = sns.color_palette('bright')
plt.pie(data, labels=keys, colors=palette_color, autopct= '%.0f%%')
plt.show();
```



CONCLUSION

The exploratory data analysis (EDA) of the *student_performance* dataset showed that student success is influenced by several factors, including *study hours, parental education, test preparation, and sleep*.

- **Study Hours**: More study time generally leads to better performance.
- **Parental Education**: Higher parental education levels correlate with improved student performance.
- **Test Preparation**: Participation in test preparation courses enhances academic outcomes.
- **Sleep**: Adequate sleep contributes positively to student performance.

Happy Learning! ✨

Thank you.