In [1]:
```
# MANISH KUMAR
```

In [2]:
```
import pandas as pd
import numpy as np
import mysql.connector
import warnings
warnings.filterwarnings("ignore")
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
df=pd.read_csv("ADMISSION.csv")
```

In [4]:
```
df
```

Out[4]:

	SL.NO	First Name	Middle Name	Last Name	Gender	DOB	Email ID	Course Name/ Standard	Admission Date	Father's Name	Mother's Name	City	State	Country
0	2	MEDHANSH	NaN	KUMAR	MALE	09.09.2021	NaN	PLAY	NaN	GOPAL KUMAR	JYOTI KUMARI	PATNA	BIHAR	INDIA
1	3	ANKUSH	NaN	NaN	MALE	10.02.2021	NaN	PLAY	NaN	CHANDAN KUMAR	GURIYA KUMARI	PATNA	BIHAR	INDIA
2	4	NAYRA	NaN	JAISWAL	FEMALE	15.10.2021	NaN	PLAY	NaN	NIRAJ JAISWAL	NISHU KUMARI	PATNA	BIHAR	INDIA
3	5	AYUSHI	NaN	NaN	FEMALE	18.09.2020	NaN	PLAY	NaN	RAHUL KUMAR	PUJA KUMARI	PATNA	BIHAR	INDIA
4	6	VEDANT	NaN	NaN	MALE	14.09.2021	NaN	PLAY	NaN	VIVEK KUMAR MANDAL	JULIE KUMARI	PATNA	BIHAR	INDIA
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
182	184	ARYAN	NaN	SINGH	MALE	10.07.2010	NaN	CLASS IX	NaN	HARENDRA KUMAR SINGH	JULI KUMARI	PATNA	BIHAR	INDIA
183	185	AFIFA	NaN	RAHMAN	FEMALE	07.10.2007	NaN	CLASS IX	NaN	ATAUR RAHMAN	NILOFAR RAHMAN	PATNA	BIHAR	INDIA
184	186	Amrit	NaN	Shekhar	MALE	12.01.2010	NaN	CLASS X	NaN	AMRENDRA CHANDRA SHEKHER	SONI SHEKHER	PATNA	BIHAR	INDIA
185	187	Kundan	NaN	Kumar	MALE	17.02.2009	NaN	CLASS X	NaN	MR. SANTOSH KUMAR	MRS. RINKU DEVI	PATNA	BIHAR	INDIA
186	188	Prince	NaN	Kumar	MALE	NaN	NaN	NaN	NaN	NaN	NaN	NaN	BIHAR	INDIA

187 rows × 16 columns

In [5]:
```
# TITLE:-"Student Enrollment Analysis for Strategic Admissions Planning"
```

In [6]:
```
# DOMAIN:- "School Dataset"
```

In [7]:
```
# DISCRIPTION:-"We looked at which classes (standards) have the most students.
              #This helps you focus on promoting the most popular courses and
              # improve marketing for those with fewer students."
```

In [8]:
```
# Skills:-      "1 Python","2 Pandas","3 Statistics","4 Data manipulation","5 Data Analysis",
#               "6 Data transformation","7 Data cleaning","8 Data visualization","9 Project methodology"
```

Business Problem Understanding

In [10]:
```
# how to students take more admission in the School
```

Data Under and Exploration

In [12]:
```
df.head()
```

Out[12]:

	SL.NO	First Name	Middle Name	Last Name	Gender	DOB	Email ID	Course Name/ Standard	Admission Date	Father's Name	Mother's Name	City	State	Country	Enro
0	2	MEDHANSH	NaN	KUMAR	MALE	09.09.2021	NaN	PLAY	NaN	GOPAL KUMAR	JYOTI KUMARI	PATNA	BIHAR	INDIA	
1	3	ANKUSH	NaN	NaN	MALE	10.02.2021	NaN	PLAY	NaN	CHANDAN KUMAR	GURIYA KUMARI	PATNA	BIHAR	INDIA	
2	4	NAYRA	NaN	JAISWAL	FEMALE	15.10.2021	NaN	PLAY	NaN	NIRAJ JAISWAL	NISHU KUMARI	PATNA	BIHAR	INDIA	
3	5	AYUSHI	NaN	NaN	FEMALE	18.09.2020	NaN	PLAY	NaN	RAHUL KUMAR	PUJA KUMARI	PATNA	BIHAR	INDIA	
4	6	VEDANT	NaN	NaN	MALE	14.09.2021	NaN	PLAY	NaN	VIVEK KUMAR MANDAL	JULIE KUMARI	PATNA	BIHAR	INDIA	

In [13]:
```
# o#serve:- by default we see top 5 rows
```

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

Out[14]:

	SL.NO	First Name	Middle Name	Last Name	Gender	DOB	Email ID	Course Name/ Standard	Admission Date	Father's Name	Mother's Name	City	State	Country	Enr
182	184	ARYAN	NaN	SINGH	MALE	10.07.2010	NaN	CLASS IX	NaN	HARENDRA KUMAR SINGH	JULI KUMARI	PATNA	BIHAR	INDIA	
183	185	AFIFA	NaN	RAHMAN	FEMALE	07.10.2007	NaN	CLASS IX	NaN	ATAUR RAHMAN	NILOFAR RAHMAN	PATNA	BIHAR	INDIA	
184	186	Amrit	NaN	Shekhar	MALE	12.01.2010	NaN	CLASS X	NaN	AMRENDRA CHANDRA SHEKHER	SONI SHEKHER	PATNA	BIHAR	INDIA	P
185	187	Kundan	NaN	Kumar	MALE	17.02.2009	NaN	CLASS X	NaN	MR. SANTOSH KUMAR	MRS. RINKU DEVI	PATNA	BIHAR	INDIA	P
186	188	Prince	NaN	Kumar	MALE	NaN	NaN	NaN	NaN	NaN	NaN	NaN	BIHAR	INDIA	P

In [15]:
```
# observe:- by default we see last 5 rows
```

In [16]:
```
df.shape
```

Out[16]: (187, 16)

In [17]:
```
# observe:-there are 187 rows and 16 columns
```

In [18]:
```
df.dtypes
```

Out[18]:
```
SL.NO                     int64
First Name               object
Middle Name              object
Last Name                object
Gender                   object
DOB                      object
Email ID                float64
Course Name/ Standard    object
Admission Date          float64
Father's Name            object
Mother's Name            object
City                     object
State                    object
Country                  object
Regn./ Enrollment No.    object
Type (Old/ New           object
dtype: object
```

In [19]:
```
# observe:- there are some categorical and continuous data types
```

In [20]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 187 entries, 0 to 186
Data columns (total 16 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   SL.NO                  187 non-null    int64  
 1   First Name             187 non-null    object 
 2   Middle Name            12 non-null     object 
 3   Last Name              160 non-null    object 
 4   Gender                 179 non-null    object 
 5   DOB                    145 non-null    object 
 6   Email ID               0 non-null      float64
 7   Course Name/ Standard  186 non-null    object 
 8   Admission Date         0 non-null      float64
 9   Father's Name          159 non-null    object 
 10  Mother's Name          156 non-null    object 
 11  City                   186 non-null    object 
 12  State                  187 non-null    object 
 13  Country                187 non-null    object 
 14  Regn./ Enrollment No.  180 non-null    object 
 15  Type (Old/ New         154 non-null    object 
dtypes: float64(2), int64(1), object(13)
memory usage: 23.5+ KB
```

In [21]:
```
# Obesreve: to show the rows ,columns,null value and data type of each columns
```

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

Out[22]:
```
SL.NO                    0
First Name               0
Middle Name            175
Last Name               27
Gender                   8
DOB                     42
Email ID               187
Course Name/ Standard    1
Admission Date         187
Father's Name           28
Mother's Name           31
City                     1
State                    0
Country                  0
Regn./ Enrollment No.    7
Type (Old/ New          33
dtype: int64
```

In [23]:
```
# observe : many columns have null value .
```

In [24]:
```
df.columns.tolist()
```

Out[24]:
```
['SL.NO',
 'First Name',
 'Middle Name',
 'Last Name',
 'Gender',
 'DOB',
 'Email ID',
 'Course Name/ Standard',
 'Admission Date',
 "Father's Name",
 "Mother's Name",
 'City',
 'State',
 'Country',
 'Regn./ Enrollment No.',
 'Type (Old/ New']
```

In [25]:
```
# Obesreve: To show all columns name in a list
# sL.NO
# Description: Serial number; a unique, sequential identifier for each record in the dataset.

# First Name
# Description: The student's given name.
```

```
# Middle Name
# Description: The student’s additional given name (optional, may be blank for some students).

# Last Name
# Description: The student’s family or surname.

# Gender
# Description: Student’s gender (e.g., Male, Female, Other).

# DOB (Date of Birth)
# Description: The student's birth date, typically in DD/MM/YYYY or YYYY-MM-DD format.

# Email ID
# Description: The student’s email address for communication.

# Course Name/ Standard
# Description: The course or grade level the student is enrolled in (e.g., "B.Sc. Computer Science", "Grade 10").

# Admission Date
# Description: The date on which the student was admitted to the institution.

# Father's Name
# Description: The full name of the student’s father.

# Mother's Name
# Description: The full name of the student’s mother.

# City
# Description: The city in which the student resides.

# State
# Description: The state or province where the student lives.

# Country
# Description: The country of the student’s residence or nationality.

# Regn./ Enrollment No.
# Description: The official registration or enrollment number assigned to the student; typically unique.

# Type (Old/ New)
# Description: Indicates whether the student is a newly admitted student or a continuing/returning (old) student.
```

In [26]:
```
df.duplicated().sum()
```

Out[26]:
```
0
```

In [27]:
```
# Obesreve: To check all rows for any duplicates values
```

explore each column wise

column 1 =SL.NO

In [30]:
```
df["SL.NO"].dtypes
```

Out[30]:
```
dtype('int64')
```

In [31]:
```
# observe:- this is the int and continuous variable and corrcet data types
```

In [32]:
```
df["SL.NO"].nunique()
```

Out[32]:
```
187
```

In [33]:
```
# each rows has unique value (if more then 30% of the rows values unique) there is no use in the analysis)
# simply drop them
```

In [34]:
```
df["SL.NO"].isnull().sum()
```

Out[34]:
```
0
```

In [35]:
```
# Observe : in this column there is no null value
```

column 2 = First Name

In [37]:
```
df["First Name"].dtypes
```

Out[37]:
```
dtype('O')
```

In [38]:
```
# Observe :- this column is discrete variable,object  Datatype and correct Datatypes
```

In [39]:
```
df["First Name"].value_counts()
```

Out[39]:
```
First Name
Aarav       3
Rishav      3
Akshat      2
Mayank      2
AKSHIT      2
           ..
AADYA       1
Aarav       1
Aadhvan     1
Aditya      1
Prince      1
Name: count, Length: 178, dtype: int64
```

In [40]:
```
# Observe : this columan has different types of value .there is no use for analysising and not any effect
# so simply remove this columns
```

In [41]:
```
df["First Name"].isnull().sum()
```

Out[41]:
```
0
```

In [42]:
```
# Observe : in this column there is no null value
```

column 3 = Middle Name

In [44]:
```
df["Middle Name"].dtypes
```

Out[44]:
```
dtype('O')
```

In [45]:
```
# Observe :- this column is discrete variable,object  Datatype and correct Datatypes
```

In [46]:
```
df["Middle Name"].value_counts().sum()
```

Out[46]:
```
12
```

In [47]:
```
# Observe : this columan has different types of value .there is no use for analysising and not any effect
# simply drop them
```

In [48]:
```
df["Middle Name"].isnull().sum()
```

Out[48]:
```
175
```

In [49]:
```
# there is large amount of null value so simply drop them
```

column 4 = Last Name

In [51]:
```
df["Last Name"].dtypes
```

Out[51]:
```
dtype('O')
```

In [52]:
```
# Observe :- this column is discrete variable,object  Datatype and correct Datatypes
```

In [53]:
```
df["Last Name"].value_counts()
```

Out[53]:
```
Last Name
Singh        19
Raj          15
Kumari       13
Kumar        12
KUMAR         8
             ..
Choudhary     1
JAISWAL       1
Shree         1
Hashmi        1
Shekhar       1
Name: count, Length: 72, dtype: int64
```

In [54]:
```
# Observe : this columan has different types of value .there is no use for analysising and not any effect
# so simply remove this columns
```

In [55]:
```
df["Last Name"].isnull().sum()
```

Out[55]:
```
27
```

In [56]:
```
# observe:-there is some  null value.
```

column 5 = Last Name

In [58]:
```
df["Gender"].dtypes
```

Out[58]:
```
dtype('O')
```

In [59]:
```
# Observe :- this column is discrete variable,object  Datatype and correct Datatypes
```

In [60]:
```
df["Gender"].value_counts().sum
```

Out[60]:
```
<bound method Series.sum of Gender
MALE      113
FEMALE     66
Name: count, dtype: int64>
```

In [61]:
```
# Observe : this columan has 2 types of value . male are more
```

In [62]:
```
df["Gender"].isnull().sum()
```

Out[62]:
```
8
```

In [63]:
```
# observe:- there is some  null value.
```

column 6 = DOB

In [65]:
```
df["DOB"].dtypes
```

Out[65]:
```
dtype('O')
```

In [66]:
```
# Observe :- this column is discrete variable,object  Datatype and correct Datatypes
```

In [67]:
```
df["DOB"].isnull().sum()
```

Out[67]:
```
42
```

In [68]:
```
# observe:- there is some  null value.
```

column 7 = Email ID

In [70]:
```
df["Email ID"].dtypes
```

Out[70]:
```
dtype('float64')
```

In [71]:
```
# Observe :- this column is  float  Datatype and incorrect Datatypes
```

In [72]:
```
df["Email ID"].isnull().sum()
```

Out[72]: 187

In [73]:
```
# there is large amount of null value so simply drop them
```

column 8 = Course Name/ Standard

In [75]:
```
# correct the name of colummn as class
```

In [76]:
```
df["Course Name/ Standard"].dtypes
```

Out[76]: dtype('O')

In [77]:
```
# Observe :-this column is discrete variable,object  Datatype and correct  Datatypes
```

In [78]:
```
df["Course Name/ Standard"].value_counts()
```

Out[78]:
```
Course Name/ Standard
CLASS II         28
CLASS 1          24
CLASS IV         21
PREP II (UKG)    17
CLASS VIII       16
CLASS VII        15
CLASS V          14
CLASS III        13
PLAY             12
PREP 1 (LKG)     11
CLASS VI         11
CLASS IX          2
CLASS X           2
Name: count, dtype: int64
```

In [79]:
```
# Observe : this columan has different types of value and class II has more.
```

column 9 = Admission Date

In [81]:
```
df["Admission Date"].dtypes
```

Out[81]: dtype('float64')

In [82]:
```
# Observe :- this column is  float  Datatype and incorrect Datatypes
```

In [83]:
```
df["Admission Date"].isnull().sum()
```

Out[83]: 187

In [84]:
```
# there is large amount of null value so simply drop them
```

In [85]:
```
#pd.set_option('display.max_rows', None)
#pd.set_option('display.max_columns', None)
```

column 10 = Father's Name

In [87]:
```
df["Father's Name"].dtypes
```

Out[87]: dtype('O')

In [88]:
```
# Observe :-this column is discrete variable,object  Datatype and correct  Datatypes
```

In [89]:
```
df["Father's Name"].isnull().sum()
```

Out[89]: 28

In [90]:
```
# observe:- there is some  null value. but there is no use in data analysis so simply drop them
```

column 11 = Mother's Name

In [92]:
```
df["Mother's Name"].dtypes
```

Out[92]: dtype('O')

In [93]:
```
# Observe :-this column is discrete variable,object  Datatype and correct  Datatypes
```

In [94]:
```
df["Mother's Name"].isnull().sum()
```

Out[94]: 31

In [95]:
```
# observe:-there is some  null value. but there is no use for analysising so simply drop this column.
```

column 12 = City

In [97]:
```
df["City"].dtypes
```

Out[97]:
```
dtype('O')
```

In [98]:
```
# Observe :-this column is discrete variable,object  Datatype and correct  Datatypes
```

In [99]:
```
df["City"].isnull().sum()
```

Out[99]:
```
1
```

In [100…
```
# observe:- there is some  null value.
```

In [101…
```
df["City"].value_counts()
```

Out[101…
```
City
PATNA    186
Name: count, dtype: int64
```

In [102…
```
# observe:- there is only one value so that there is no use in data analytics  drop them
```

column 13 = State

In [104…
```
df["State"].dtypes
```

Out[104…
```
dtype('O')
```

In [105…
```
# Observe :-this column is discrete variable,object  Datatype and correct  Datatypes
```

In [106…
```
df["State"].isnull().sum()
```

Out[106…
```
0
```

In [107…
```
# observe:- there is no null value.
```

In [108…
```
df["State"].value_counts()
```

Out[108…
```
State
BIHAR     187
Name: count, dtype: int64
```

In [109…
```
# observe:- there is only one value so that there is no use in data analytics  drop them
```

column 14 = Country

In [111…
```
df["Country"].dtypes
```

Out[111…
```
dtype('O')
```

In [112…
```
# Observe :-this column is discrete variable,object  Datatype and correct  Datatypes
```

In [113…
```
df["Country"].value_counts()
```

Out[113…
```
Country
INDIA    187
Name: count, dtype: int64
```

In [114…
```
# observe:- there is only one value so that there is no use in data analytics  drop them
```

column 15 = Regn./ Enrollment No.

In [116…
```
df["Regn./ Enrollment No."].dtypes
```

Out[116…
```
dtype('O')
```

In [117…
```
# Observe :-this column is discrete variable,object  Datatype and correct  Datatypes
```

In [118…
```
df["Regn./ Enrollment No."].nunique()
```

Out[118…
```
179
```

In [119…
```
# observe:- there is different value but there is no use in data analytics so that drop them
```

column 16 = Type (Old/ New

In [121…
```
df["Type (Old/ New"].dtypes
```

Out[121…
```
dtype('O')
```

In [122…
```
# Observe :-this column is discrete variable,object  Datatype and correct  Datatypes
```

In [123…
```
df["Type (Old/ New"].value_counts()
```

Out[123…
```
Type (Old/ New
OLD    118
NEW     36
Name: count, dtype: int64
```

In [124…
```
# observe:- there is only two type of value but there is  use in data analytics
```

Data Preprocessing

In [126…
```
#1>> Data cleaning
```

In [127…
```
# reasion:- columns name change as per required
df.rename(columns={"Course Name/ Standard":"Class"},inplace=True)
```

In [128…
```
# calculate the age and create new column name "age"
#today = pd.Timestamp.today()
#df['Age'] = df['DOB'].apply(lambda x: today.year - x.year if pd.notnull(x) else None)
```

In [129…
```
# treat wrong data  (there is a wrong data)
```

In [130…
```
df["Type (Old/ New"]=df["Type (Old/ New"].replace({"New":"NEW"})
```

In [131…
```
# treat wrong data type
```

In [132…
```
df['DOB'] = pd.to_datetime(df['DOB'], errors='coerce')
```

In [133…
```
# reason:-(there is a wrong data type)
```

In [134…
```
# treat duplicates (there is no duplicate)
```

In [135…
```
# treat outlier (there is no int data type for finding outliers)
```

In [136…
```
# treat missing value
```

In [137…
```
a=df["Gender"].mode()[0]
```

In [138…
```
df["Gender"].fillna(a,inplace=True)
```

In [139…
```
# reasion:- If most students are one gender (e.g., Male), fill missing values with mode.
```

In [140…
```
df.drop(index=186,inplace=True)
```

In [141…
```
#reasion:-there is no option for filling the class because this columns is effect on anaysising so simply remove
```

In [142…
```
df['Type (Old/ New'] = df['Type (Old/ New'].fillna('NEW')
```

In [143…
```
#reasion:-If unsure or for very sensitive reporting, use "Unknown" instead — but "New" is the most logical and
          #useful guess in an admissions context.
```

In [ ]:
```

```

In [144…
```
# 2>>dimension reduction
```

In [145…
```
df.drop(columns=["SL.NO","First Name","Middle Name","Last Name","Email ID","Admission Date"
                 ,"Father's Name","Mother's Name","City","State","Country",
                "Regn./ Enrollment No."],inplace=True)
```

In [146…
```
# reasion:- there is no use of this columns for analysising
```

In [147…
```
df["DOB"].isnull().sum()
```

Out[147…
```
140
```

In [148…
```
df.drop(columns="DOB",inplace=True)
```

In [149…
```
# reasion:- there is large amount of null value so that simply remove them.
```

In [150…
```
df["Type (Old/ New"].mode()
```

Out[150…
```
0    OLD
Name: Type (Old/ New, dtype: object
```

In [151…
```
df
```

Out[151…

	Gender	Class	Type (Old/ New
0	MALE	PLAY	NEW
1	MALE	PLAY	NEW
2	FEMALE	PLAY	NEW
3	FEMALE	PLAY	NEW
4	MALE	PLAY	NEW
...	...	...	...
181	MALE	CLASS VIII	NEW
182	MALE	CLASS IX	NEW
183	FEMALE	CLASS IX	NEW
184	MALE	CLASS X	OLD
185	MALE	CLASS X	OLD

186 rows × 3 columns

Data Analysis and Visualization

In [153…
```
# notes:- there is only three column are imortant (     Gender, Class    ,Type (Old/ New))
```

In [154…
```
#Total number of students in the school that takes dmission
df["Gender"].value_counts().sum()
```

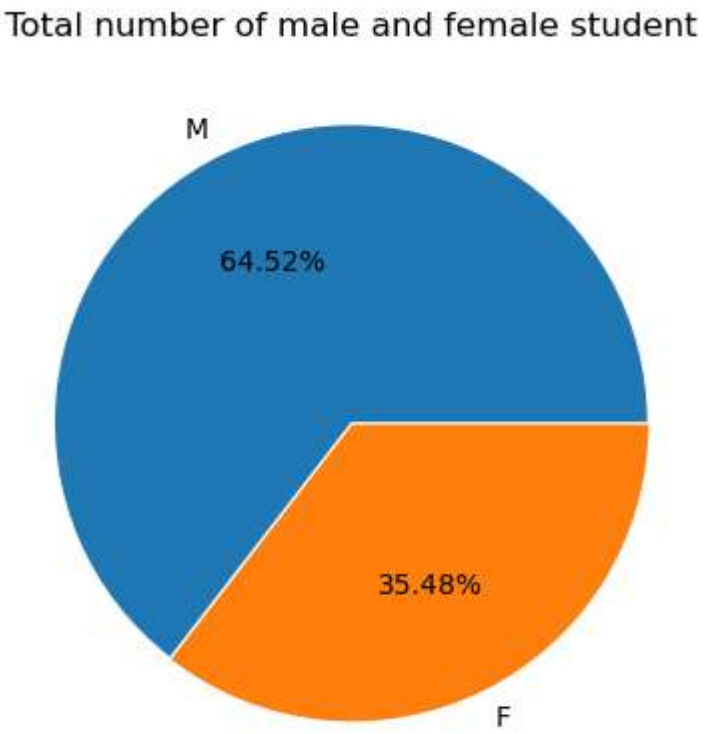
Out[154… 186

In [155…
```
# how many male and female students take admission
df["Gender"].value_counts()
```

Out[155…
```
Gender
MALE      120
FEMALE     66
Name: count, dtype: int64
```

In [156…
```
plt.pie(df["Gender"].value_counts(),autopct="%1.2f%%",labels=["M","F"],explode=[0,0.01])
plt.title("Total number of male and female student")
```

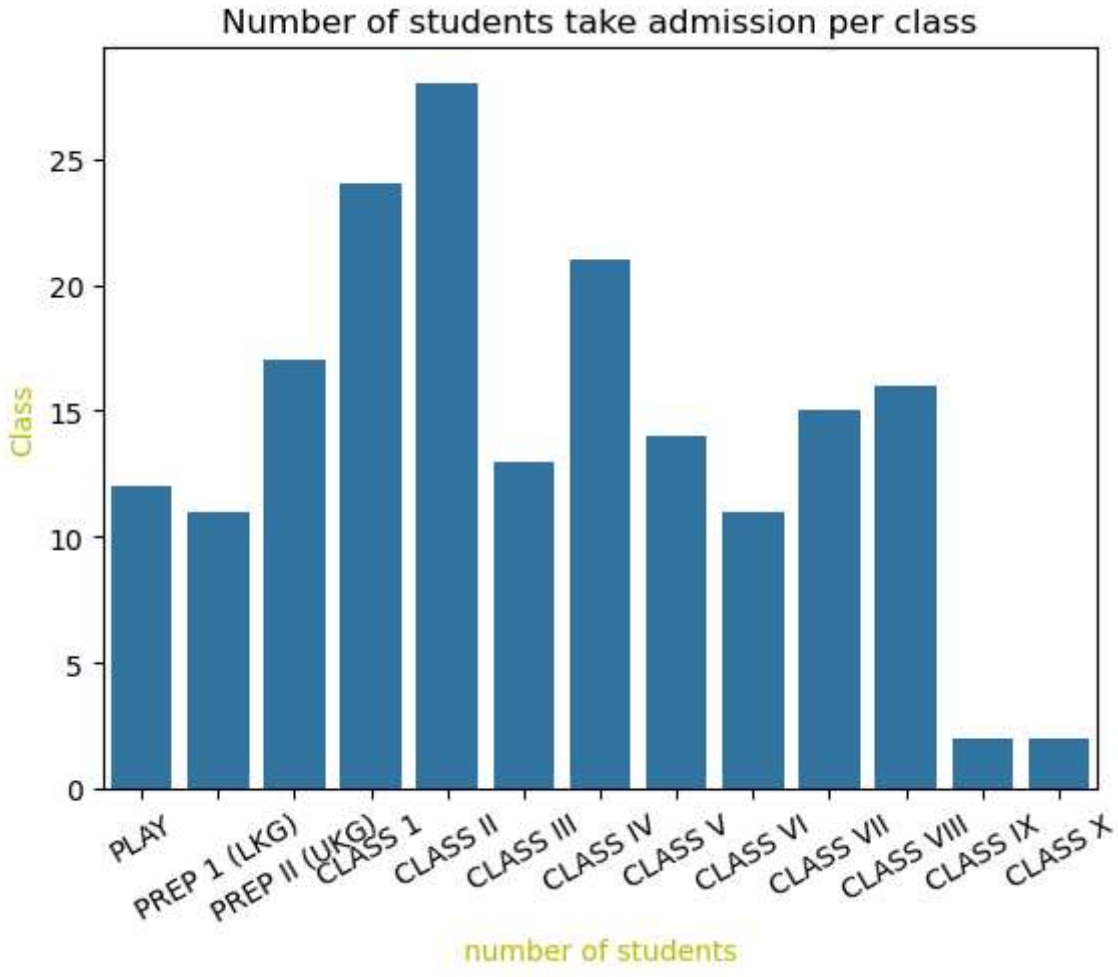
```
plt.savefig("male and female student.png")
plt.show()
```



In [157…
```
# Number of students take admission per class
df["Class"].value_counts()
```

Out[157…
```
Class
CLASS II         28
CLASS 1          24
CLASS IV         21
PREP II (UKG)    17
CLASS VIII       16
CLASS VII        15
CLASS V          14
CLASS III        13
PLAY             12
PREP 1 (LKG)     11
CLASS VI         11
CLASS IX          2
CLASS X           2
Name: count, dtype: int64
```

In [158…]
```
sns.countplot(x=df["Class"])
plt.xlabel("number of students",size=10,c="y")
plt.ylabel("Class",size=10,c="y")
plt.xticks(rotation=30)
plt.title("Number of students take admission per class")
plt.savefig("student per calsss.png")
plt.show()
```



In [159…]
```
# class wise admission takes male or female
pd.crosstab(df["Class"],df["Gender"])
```

Out[159…

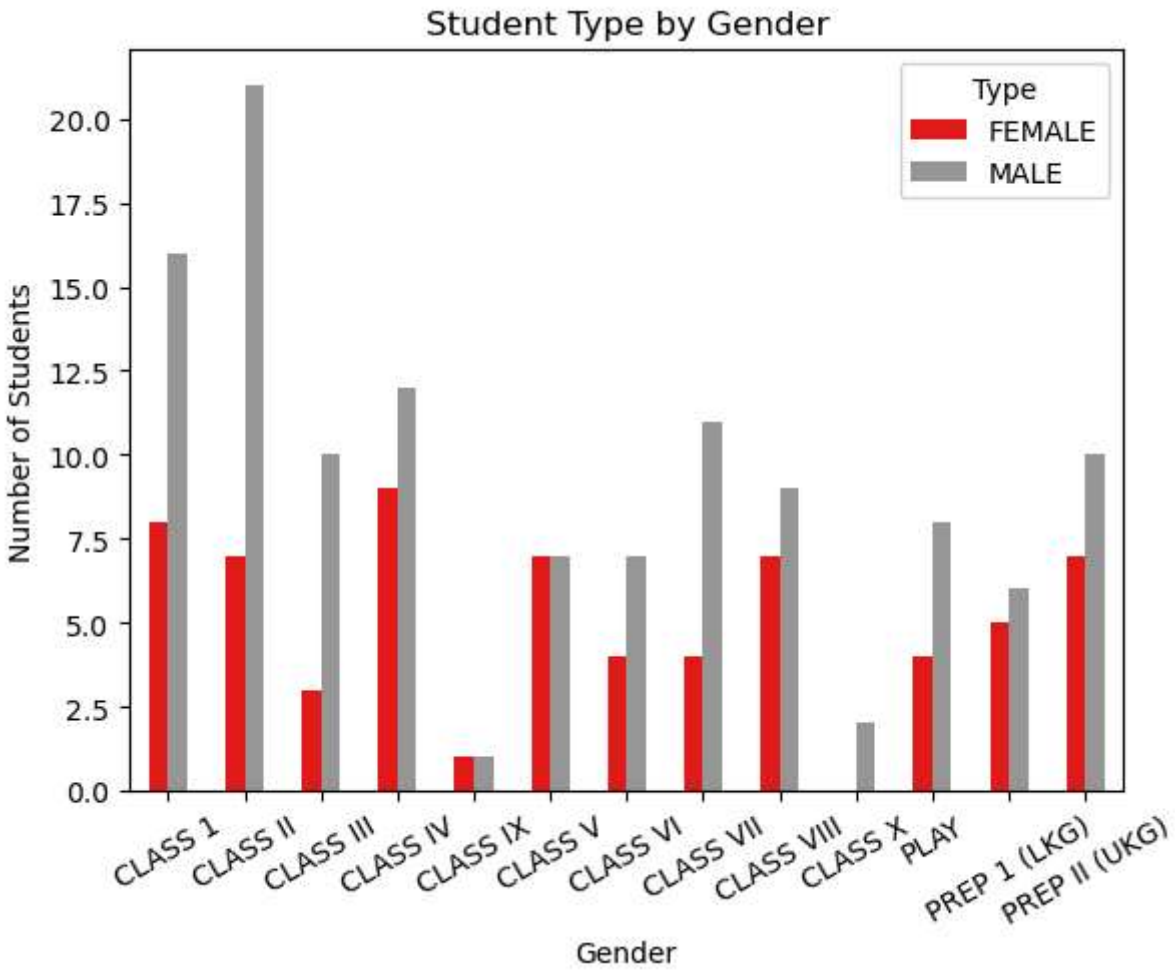
Gender	FEMALE	MALE
Class		
CLASS 1	8	16
CLASS II	7	21
CLASS III	3	10
CLASS IV	9	12
CLASS IX	1	1
CLASS V	7	7
CLASS VI	4	7
CLASS VII	4	11
CLASS VIII	7	9
CLASS X	0	2
PLAY	4	8
PREP 1 (LKG)	5	6
PREP II (UKG)	7	10

In [160…

```
cg=pd.crosstab(df["Class"],df["Gender"])
```

In [161…

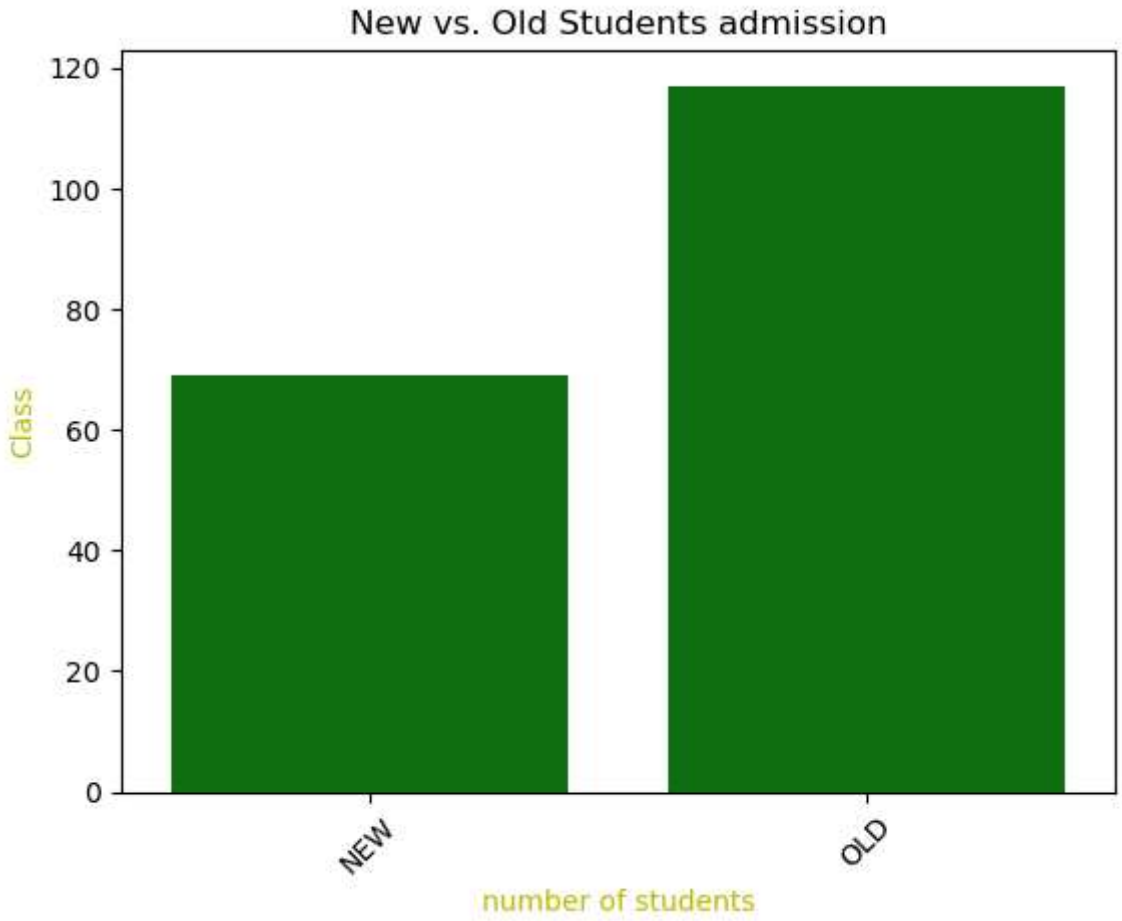
```
cg.plot(kind='bar', stacked=False, colormap='Set1')
plt.title("Student Type by Gender")
plt.ylabel("Number of Students")
plt.xlabel("Gender")
plt.xticks(rotation=30)
plt.legend(title="Type")
plt.savefig("male and female student classs.png")
plt.show()
```



In [162…
```
# New vs. Old Students admission
df["Type (Old/ New"].value_counts()
```

Out[162…
```
Type (Old/ New
OLD    117
NEW     69
Name: count, dtype: int64
```

In [163…
```
sns.countplot(x=df["Type (Old/ New"],color="g")
plt.xlabel("number of students",size=10,c="y")
plt.ylabel("Class",size=10,c="y")
plt.xticks(rotation=45)
plt.title("New vs. Old Students admission")
plt.savefig("student admission.png")
plt.show()
```



In [164…

```
# class wise new and old student  admission
pd.crosstab(df["Class"],df["Type (Old/ New"])
```

Out[164…

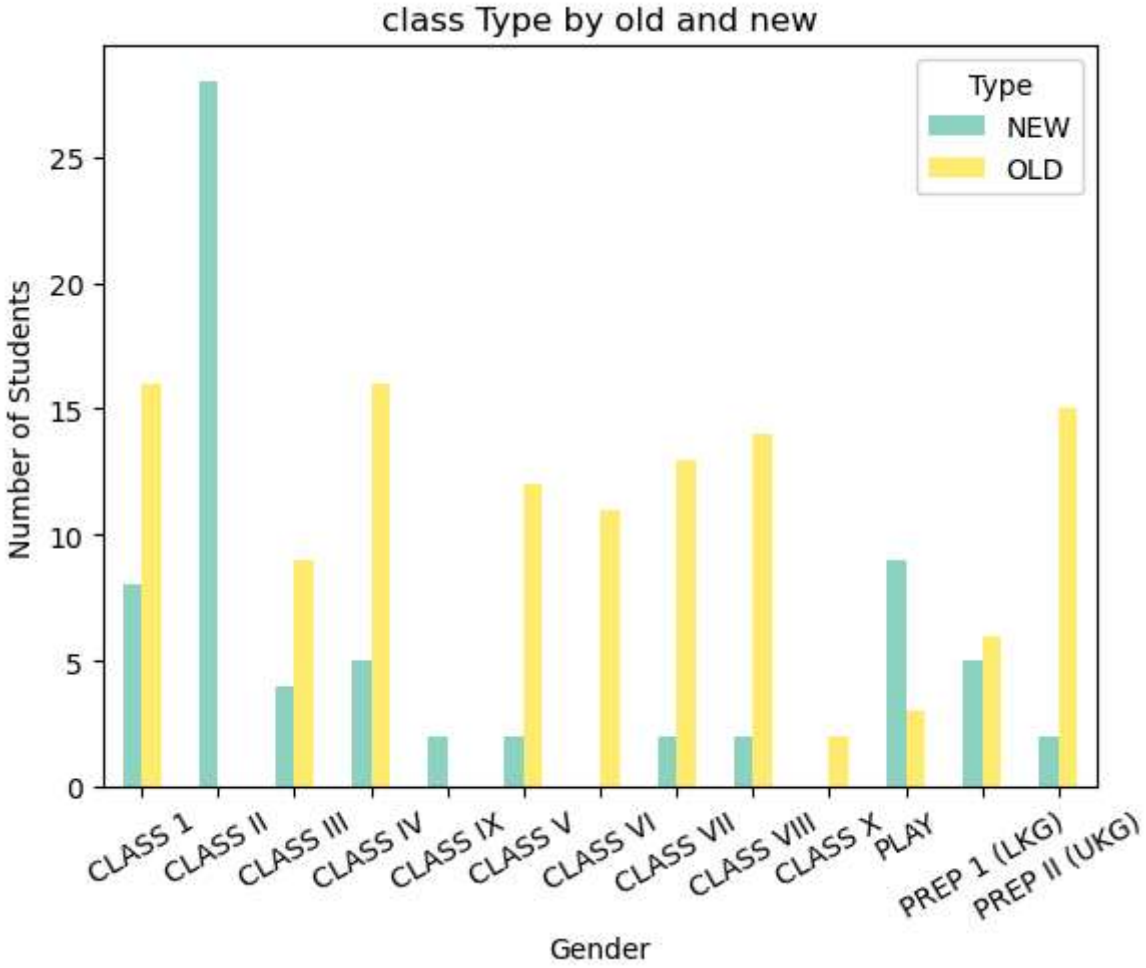
Type (Old/ New Class	NEW	OLD
CLASS 1	8	16
CLASS II	28	0
CLASS III	4	9
CLASS IV	5	16
CLASS IX	2	0
CLASS V	2	12
CLASS VI	0	11
CLASS VII	2	13
CLASS VIII	2	14
CLASS X	0	2
PLAY	9	3
PREP 1 (LKG)	5	6
PREP II (UKG)	2	15

In [165…

```
con=pd.crosstab(df["Class"],df["Type (Old/ New"])
```

In [166…

```
con.plot(kind='bar', stacked=False, colormap='Set3')
plt.title("class Type by old and new")
plt.ylabel("Number of Students")
plt.xlabel("Gender")
plt.xticks(rotation=30)
plt.legend(title="Type")
plt.savefig("class students.png")
plt.show()
```



In [167…
```
# class wise female and male student  admission
pd.crosstab(df["Gender"],df["Type (Old/ New"])
```

Out[167…

Type (Old/ New	NEW	OLD
Gender		
FEMALE	18	48
MALE	51	69

In [168…
```
ct=pd.crosstab(df["Gender"],df["Type (Old/ New"])
```

In [169…
```
ct.plot(kind='bar', stacked=False, colormap='Set1')
plt.title("Student Type by Gender")
plt.ylabel("Number of Students")
```

```
plt.xlabel("Gender")
plt.xticks(rotation=45)
plt.legend(title="Type")
plt.savefig("male and female oln student.png")
plt.show()
```