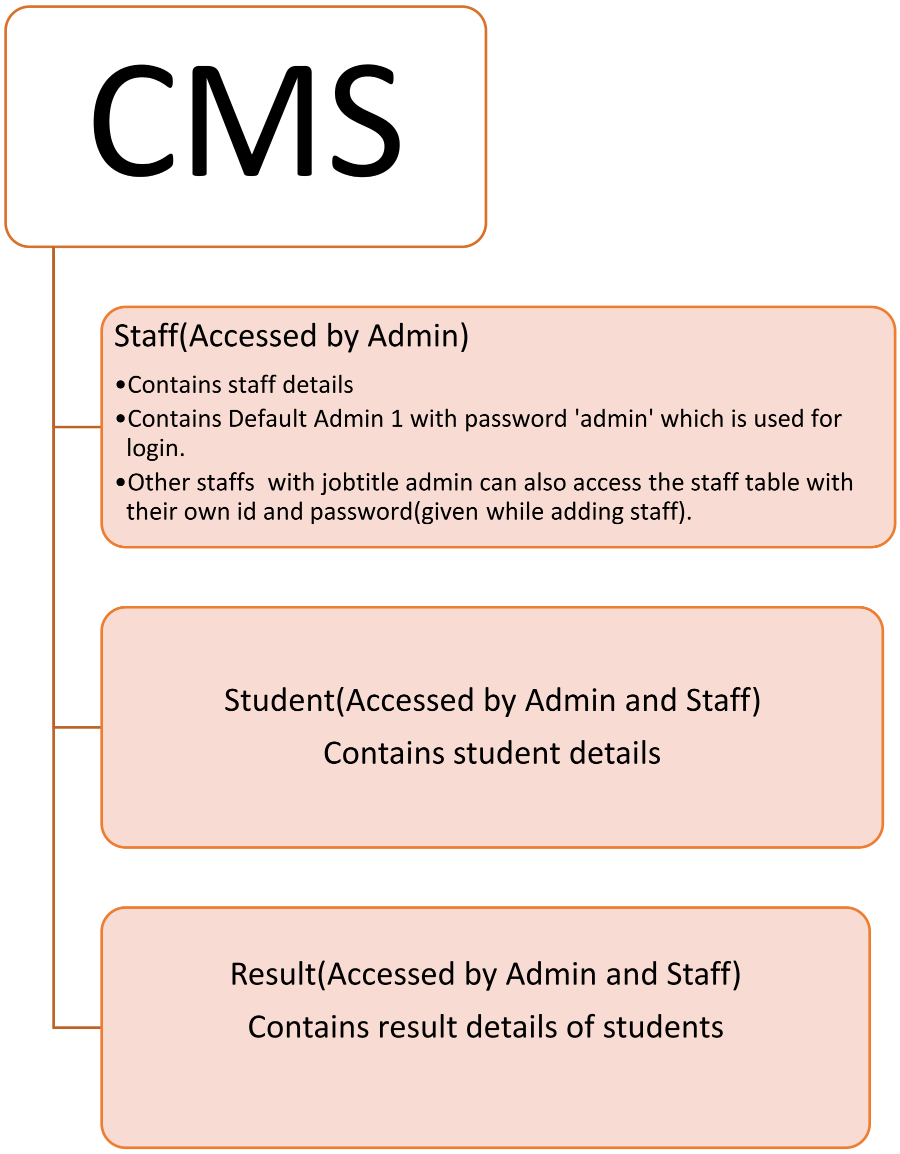
College Management System



CMS

Staff(Accessed by Admin)

- Contains staff details
- Contains Default Admin 1 with password 'admin' which is used for login.
- Other staffs with jobtitle admin can also access the staff table with their own id and password(given while adding staff).

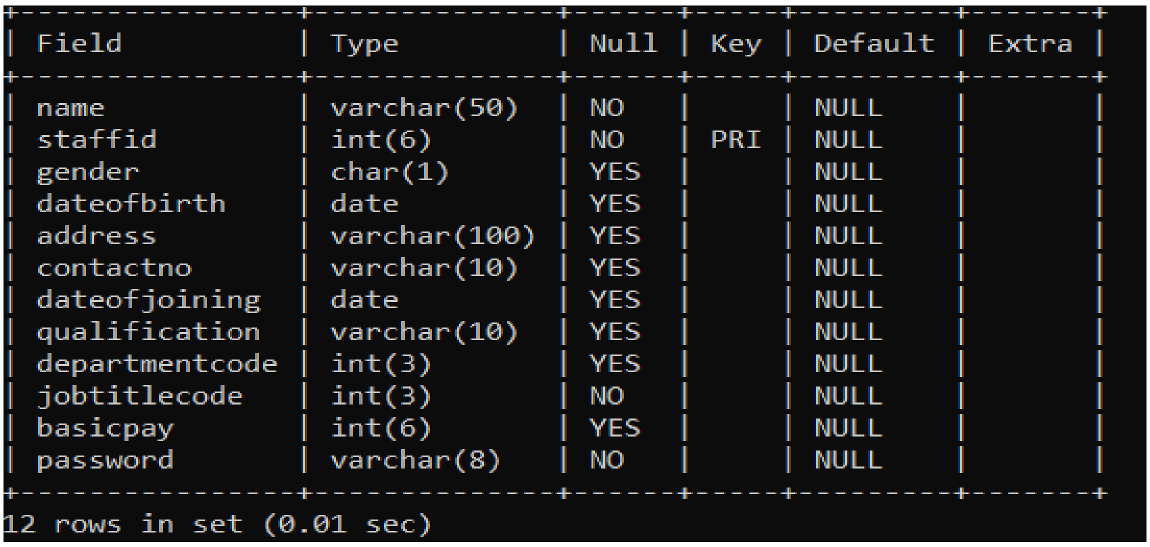
Student(Accessed by Admin and Staff)

Contains student details

Result(Accessed by Admin and Staff)

Contains result details of students

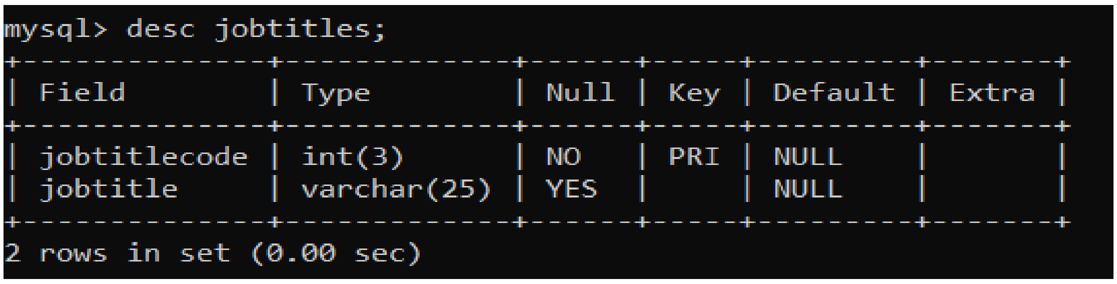
Staff table



```
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| name           | varchar(50)  | NO   |     | NULL    |       |
| staffid        | int(6)       | NO   | PRI | NULL    |       |
| gender         | char(1)      | YES  |     | NULL    |       |
| dateofbirth    | date         | YES  |     | NULL    |       |
| address        | varchar(100) | YES  |     | NULL    |       |
| contactno      | varchar(10)  | YES  |     | NULL    |       |
| dateofjoining  | date         | YES  |     | NULL    |       |
| qualification  | varchar(10)  | YES  |     | NULL    |       |
| departmentcode | int(3)       | YES  |     | NULL    |       |
| jobtitlecode   | int(3)       | NO   |     | NULL    |       |
| basicpay       | int(6)       | YES  |     | NULL    |       |
| password       | varchar(8)   | NO   |     | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
12 rows in set (0.01 sec)
```

Field	Points to be remembered
Name	Can contain Alphabets, Full stop and space. Cannot contain numbers or any other special characters
Staff id	Auto generated. Selecting the maximum among the staff ids that already exist and incrementing it by 1 to get a new id.
Gender	Only (M/F) is allowed
Date of birth	Current year-Birth year>22. i.e minimum age=22 years. Birth year cannot be less than 1930 as a staff cannot be 91 years old.
Date of joining	Dob cannot be equal to or greater than Doj.
Qualification	Can contain only alphabets. No numbers allowed.(Maximum length=10)

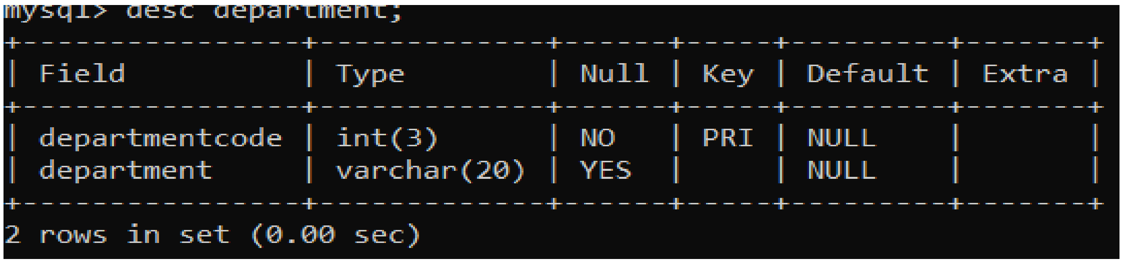
Address	Maximum length=100,Minimum length=10
Contact no	Exactly 10 numbers. No letters permitted.
Department code	Choices for department code will be made available from the department table while adding a staff and the corresponding department code needs to be entered.
Job title code	Choices for job title code will be made available from the job titles table while adding a staff and the corresponding job title code needs to be entered.
Basic Pay	Minimum basic pay=10000
Password	Minimum length=5 Maximum length=8 At least one special character and number needs to be present.



```
mysql> desc jobtitles;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| jobtitlecode | int(3)      | NO   | PRI | NULL    |       |
| jobtitle     | varchar(25) | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Job titles table is used to display the choice of job titles and the corresponding job title code so that the user can enter the appropriate job title code .This has been done to ensure the uniformity in all entries.

Field	Points to be remembered
jobtitlecode	Autogenerated. Maximum of the available job title code is selected and is incremented by 1.
jobtitle	‘Admin ’ is the job title that exists by default and has a jobtitecode=1.A job title can be added to the table only if the job title does not exist in the table already.


```
mysql> desc department;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| departmentcode | int(3)      | NO   | PRI | NULL    |       |
| department     | varchar(20) | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Department table is used to display the choice of department and the corresponding department code so that the user can enter the appropriate department code. This has been done to ensure the uniformity in all entries.

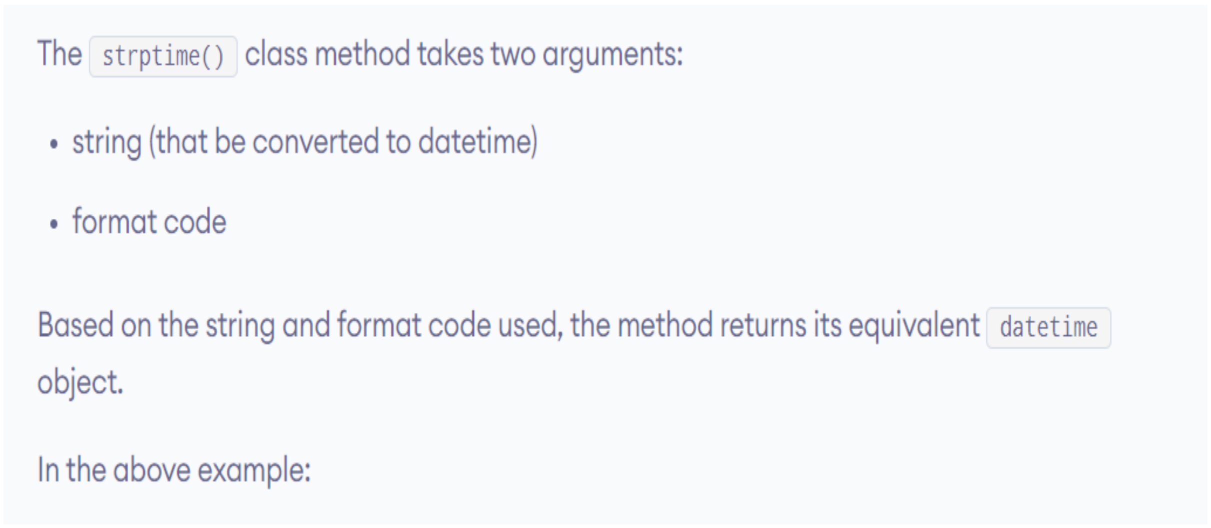
Field	Points to be remembered
departmentcode	Autogenerated. Maximum of the available department code is selected and is incremented by 1.
department	‘NA’ (“not assigned”) is the department that exists by default and has a department code=1.A department can be added to the table only if the

	department does not exist in the table already.

Date Validation:

The `strptime()` method creates a datetime object from the given string.

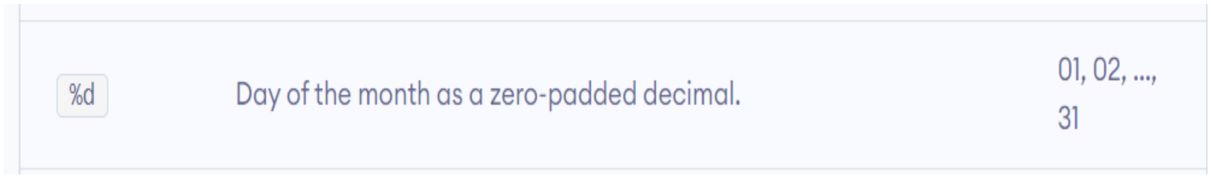
Note: You cannot create `datetime` object from every string. The string needs to be in a certain format.


The `strptime()` class method takes two arguments:

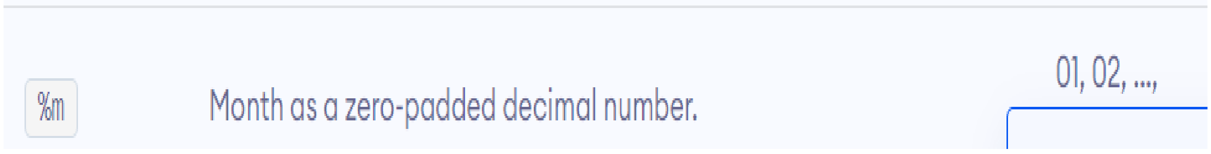
- string (that be converted to datetime)
- format code

Based on the string and format code used, the method returns its equivalent `datetime` object.

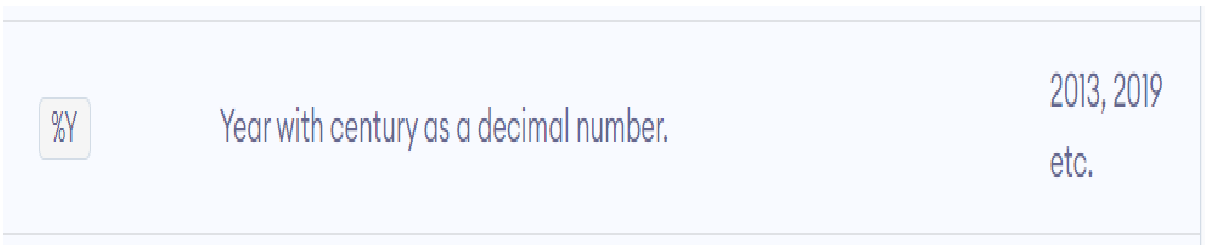
In the above example:



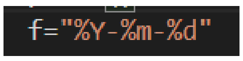
`%d`	Day of the month as a zero-padded decimal.	01, 02, ..., 31



`%m`	Month as a zero-padded decimal number.	01, 02, ...,

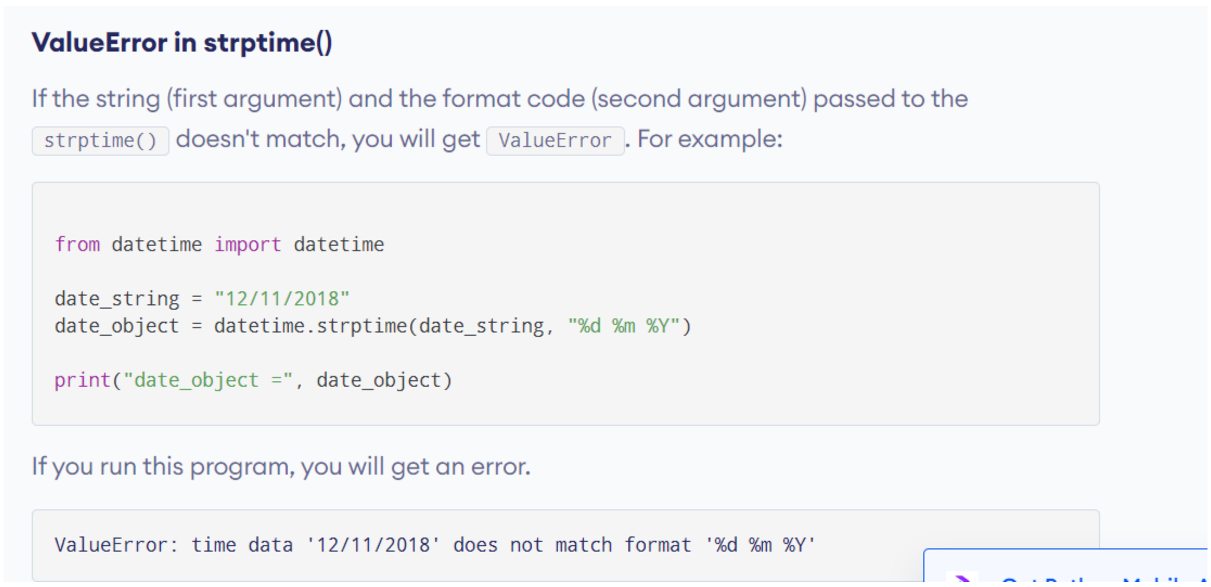


`%Y`	Year with century as a decimal number.	2013, 2019 etc.



```
f="%Y-%m-%d"
```

Format of date used in our project is ‘YYYY-MM-DD’



ValueError in strptime()

If the string (first argument) and the format code (second argument) passed to the `strptime()` doesn't match, you will get `ValueError`. For example:

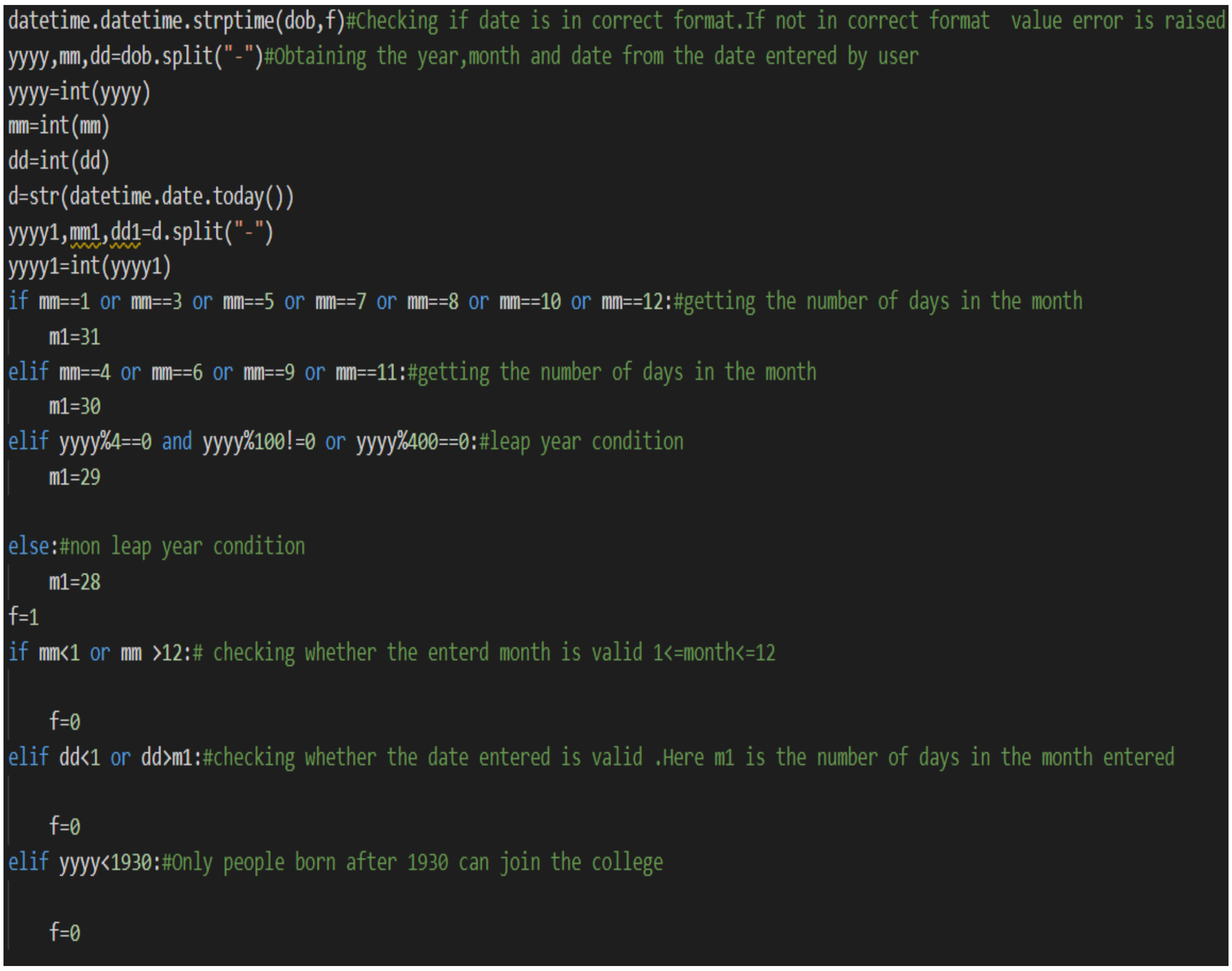
```
from datetime import datetime

date_string = "12/11/2018"
date_object = datetime.strptime(date_string, "%d %m %Y")

print("date_object =", date_object)
```

If you run this program, you will get an error.

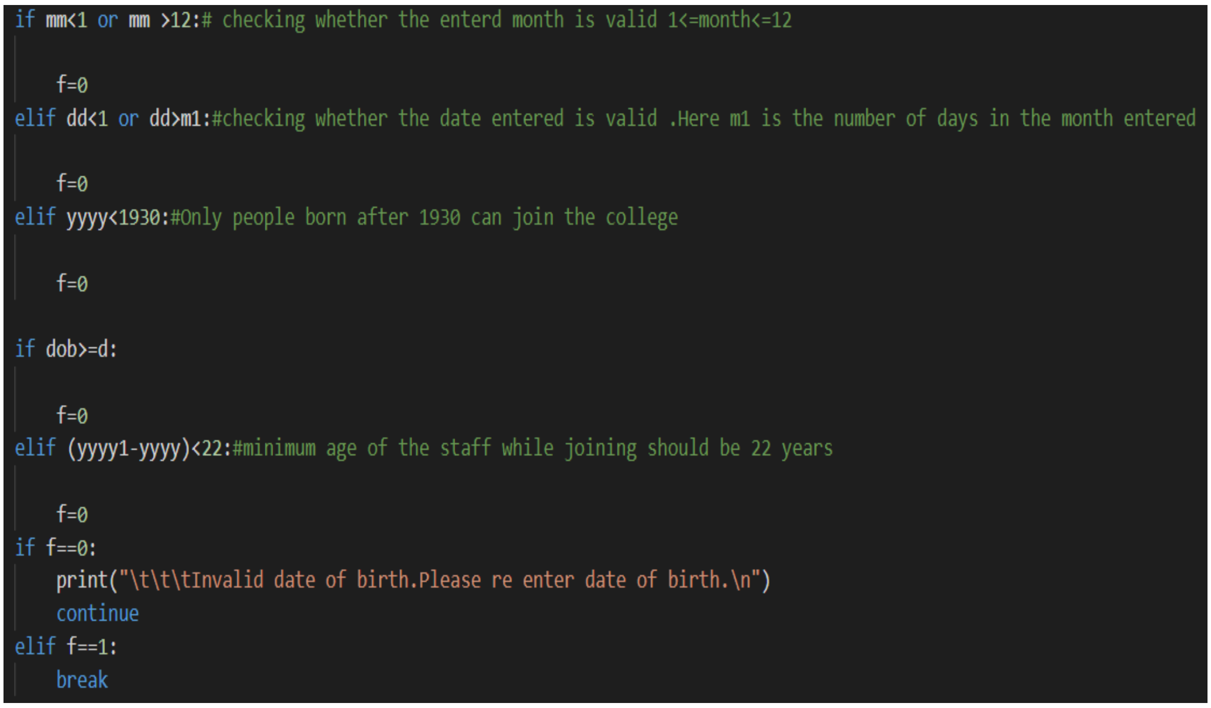
```
ValueError: time data '12/11/2018' does not match format '%d %m %Y'
```

Date validation code. Refer comments in the code for the purpose of every statements.

For date of birth:

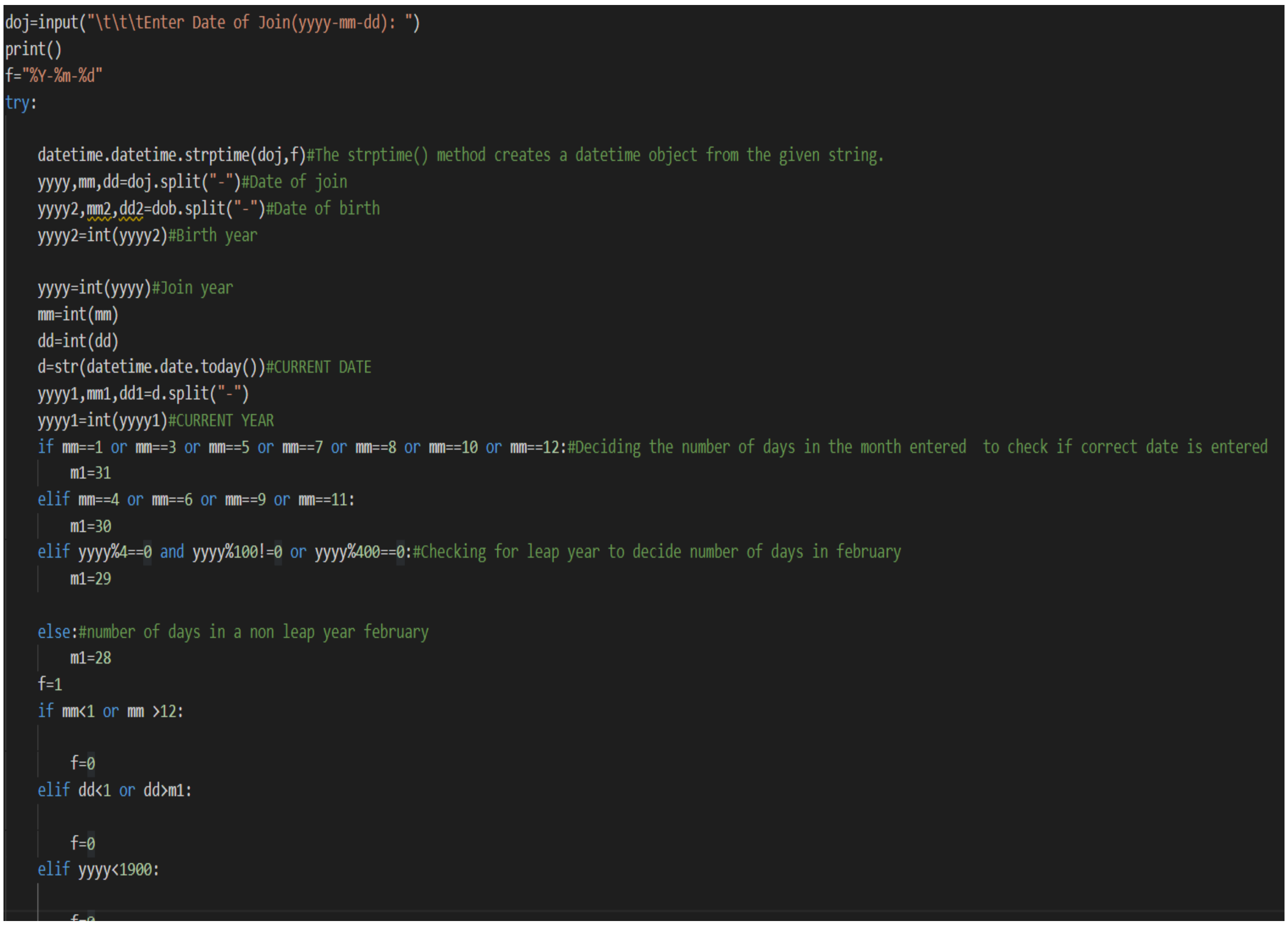


```
datetime.datetime.strptime(dob,f)#Checking if date is in correct format.If not in correct format  value error is raised
yyyy,mm,dd=dob.split("-")#Obtaining the year,month and date from the date entered by user
yyyy=int(yyyy)
mm=int(mm)
dd=int(dd)
d=str(datetime.date.today())
yyyy1,mm1,dd1=d.split("-")
yyyy1=int(yyyy1)
if mm==1 or mm==3 or mm==5 or mm==7 or mm==8 or mm==10 or mm==12:#getting the number of days in the month
    m1=31
elif mm==4 or mm==6 or mm==9 or mm==11:#getting the number of days in the month
    m1=30
elif yyyy%4==0 and yyyy%100!=0 or yyyy%400==0:#leap year condition
    m1=29

else:#non leap year condition
    m1=28
f=1
if mm<1 or mm >12:# checking whether the enterd month is valid 1<=month<=12

    f=0
elif dd<1 or dd>m1:#checking whether the date entered is valid .Here m1 is the number of days in the month entered

    f=0
elif yyyy<1930:#Only people born after 1930 can join the college

    f=0
```

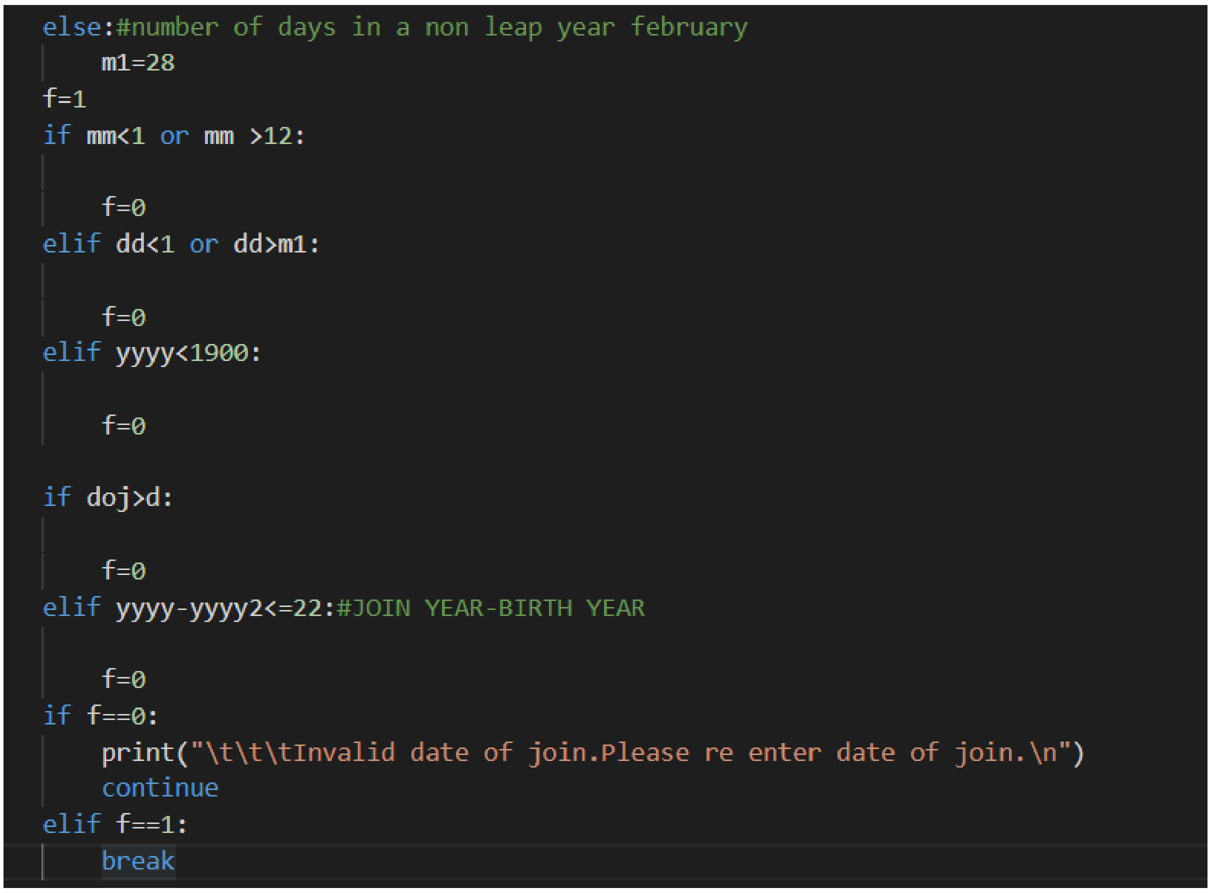


```python
if mm<1 or mm >12:# checking whether the enterd month is valid 1<=month<=12

    f=0
elif dd<1 or dd>m1:#checking whether the date entered is valid .Here m1 is the number of days in the month entered

    f=0
elif yyyy<1930:#Only people born after 1930 can join the college

    f=0

if dob>=d:

    f=0
elif (yyyy1-yyyy)<22:#minimum age of the staff while joining should be 22 years

    f=0
if f==0:
    print("\t\t\tInvalid date of birth.Please re enter date of birth.\n")
    continue
elif f==1:
    break
```

For Date of join:

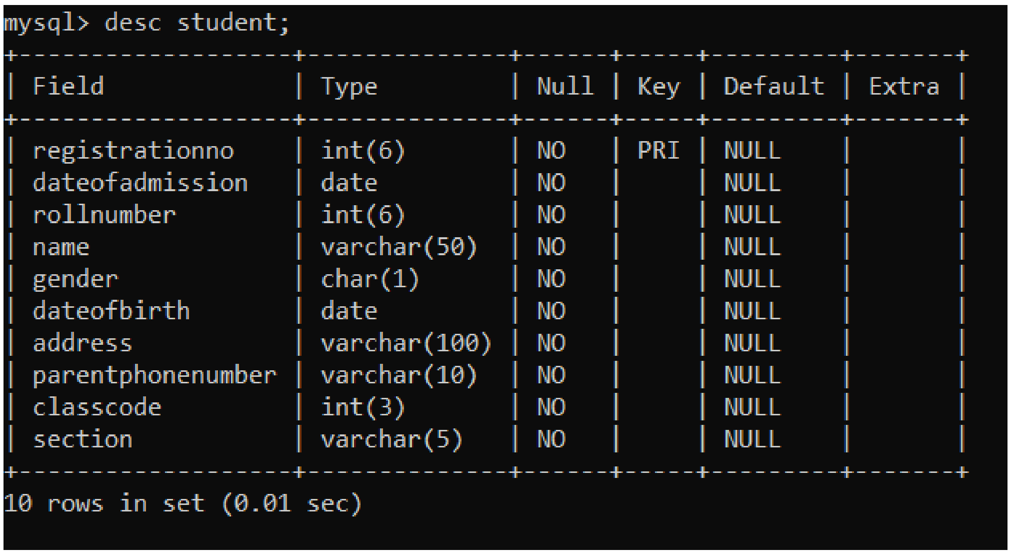


```python
doj=input("\t\t\tEnter Date of Join(yyyy-mm-dd): ")
print()
f="%Y-%m-%d"
try:

    datetime.datetime.strptime(doj,f)#The strptime() method creates a datetime object from the given string.
    yyyy,mm,dd=doj.split("-")#Date of join
    yyyy2,mm2,dd2=dob.split("-")#Date of birth
    yyyy2=int(yyyy2)#Birth year

    yyyy=int(yyyy)#Join year
    mm=int(mm)
    dd=int(dd)
    d=str(datetime.date.today())#CURRENT DATE
    yyyy1,mm1,dd1=d.split("-")
    yyyy1=int(yyyy1)#CURRENT YEAR
    if mm==1 or mm==3 or mm==5 or mm==7 or mm==8 or mm==10 or mm==12:#Deciding the number of days in the month entered  to check if correct date is entered
        m1=31
    elif mm==4 or mm==6 or mm==9 or mm==11:
        m1=30
    elif yyyy%4==0 and yyyy%100!=0 or yyyy%400==0:#Checking for leap year to decide number of days in february
        m1=29

    else:#number of days in a non leap year february
        m1=28
    f=1
    if mm<1 or mm >12:

        f=0
    elif dd<1 or dd>m1:

        f=0
    elif yyyy<1900:

        f=0
```



```python
else:#number of days in a non leap year february
    m1=28
f=1
if mm<1 or mm >12:

    f=0
elif dd<1 or dd>m1:

    f=0
elif yyyy<1900:

    f=0

if doj>d:

    f=0
elif yyyy-yyyy2<=22:#JOIN YEAR-BIRTH YEAR

    f=0
if f==0:
    print("\t\t\tInvalid date of join.Please re enter date of join.\n")
    continue
elif f==1:
    break
```

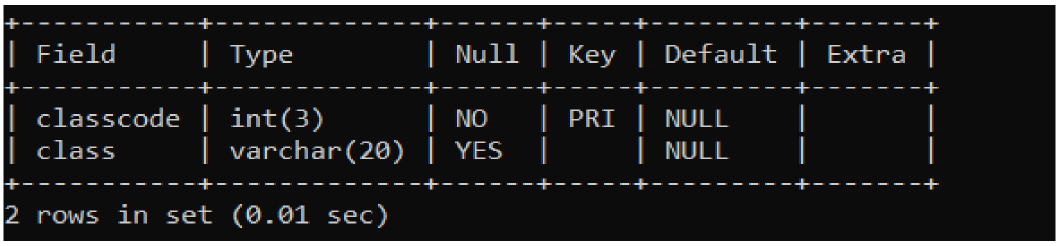
Student table



```
mysql> desc student;
+-------------------+--------------+------+-----+---------+-------+
| Field             | Type         | Null | Key | Default | Extra |
+-------------------+--------------+------+-----+---------+-------+
| registrationno    | int(6)       | NO   | PRI | NULL    |       |
| dateofadmission   | date         | NO   |     | NULL    |       |
| rollnumber        | int(6)       | NO   |     | NULL    |       |
| name              | varchar(50)  | NO   |     | NULL    |       |
| gender            | char(1)      | NO   |     | NULL    |       |
| dateofbirth       | date         | NO   |     | NULL    |       |
| address           | varchar(100) | NO   |     | NULL    |       |
| parentphonenumber | varchar(10)  | NO   |     | NULL    |       |
| classcode         | int(3)       | NO   |     | NULL    |       |
| section           | varchar(5)   | NO   |     | NULL    |       |
+-------------------+--------------+------+-----+---------+-------+
10 rows in set (0.01 sec)
```

Field	Points to be remembered
Registrationno	Auto generated. Maximum of the existing registration numbers is obtained and is incremented by 1 so as to obtain the new registration number.
Date of admission	Minimum age of student=18
Rollnumber	Can contain only numbers. Should be unique for every student.
Name	Can contain only alphabets, full stop and space. Numbers and other special characters are not allowed.
Gender	Only M/F are allowed
Date of birth	Minimum age of student=18
Address	Maximum length=100,Minimum length=10
Parent phone number	Can contain exactly 10 digits. Alphabets and special characters are not allowed.

Classcode	Choices for class code will be made available from the class table while adding a student and the corresponding class code needs to be entered. Eg:Civil,Mechanical,CSE
Section	A- First year B- Second Year C- Third year D- Fourth year



```
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| classcode | int(3)      | NO   | PRI | NULL    |       |
| class     | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```

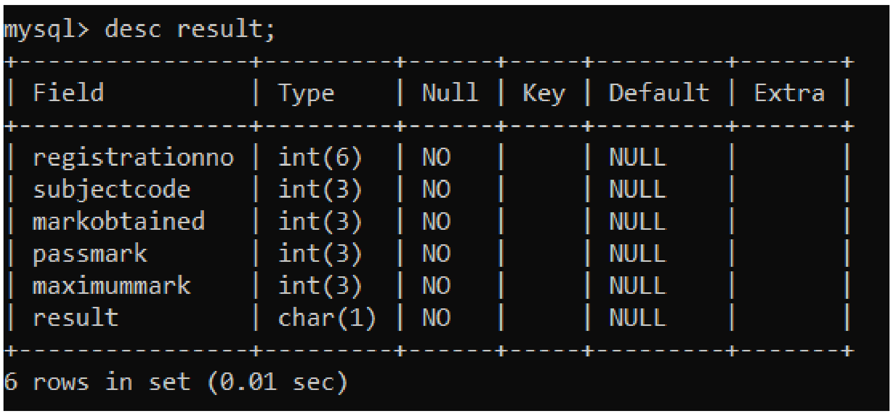
Class table is used to display the choice of class and the corresponding class code so that the user can enter the appropriate class code. This has been done to ensure the uniformity in all entries.

Note: Students can be added to the student table only if class table has at least one class.

Field	Points to be remembered
classcode	Autogenerated. Maximum of the available class code is selected and is incremented by 1.
class	A class can be added to the table only if the class does not exist in the table already.

For date validation refer code and corresponding comments.

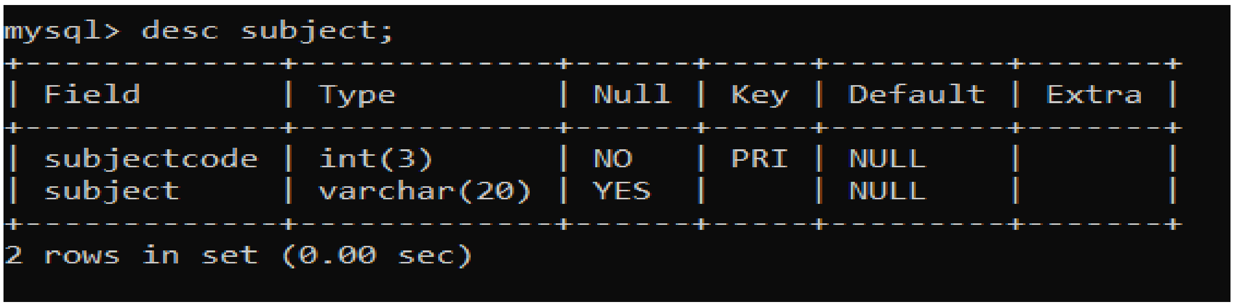
RESULT TABLE



```
mysql> desc result;
+----------------+---------+------+-----+---------+-------+
| Field          | Type    | Null | Key | Default | Extra |
+----------------+---------+------+-----+---------+-------+
| registrationno | int(6)  | NO   |     | NULL    |       |
| subjectcode    | int(3)  | NO   |     | NULL    |       |
| markobtained   | int(3)  | NO   |     | NULL    |       |
| passmark       | int(3)  | NO   |     | NULL    |       |
| maximummark    | int(3)  | NO   |     | NULL    |       |
| result         | char(1) | NO   |     | NULL    |       |
+----------------+---------+------+-----+---------+-------+
6 rows in set (0.01 sec)
```

Field	Points to remember
Registrationno	Should exist in student table
Subjectcode	Choices for subject code will be made available from the subject table while adding a result and the corresponding subject code needs to be entered.
Markobtained	Mark obtained cannot be greater than maximum mark
Passmark	Pass mark cannot be greater than maximum mark.
Maximummark	Maximum mark have to be greater than pass mark and mark obtained.
Result	P-Pass F-Fail A-Absent

Duplicate results that is a result with the same registration number and subject code is not accepted.



```
mysql> desc subject;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| subjectcode | int(3)      | NO   | PRI | NULL    |       |
| subject     | varchar(20) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

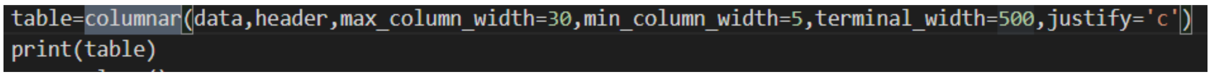
Subject table is used to display the choice of subject and the corresponding subject code so that the user can enter the appropriate subject code. This has been done to ensure the uniformity in all entries.

Note: Results can be added to the result table only if subject table has at least one subject.

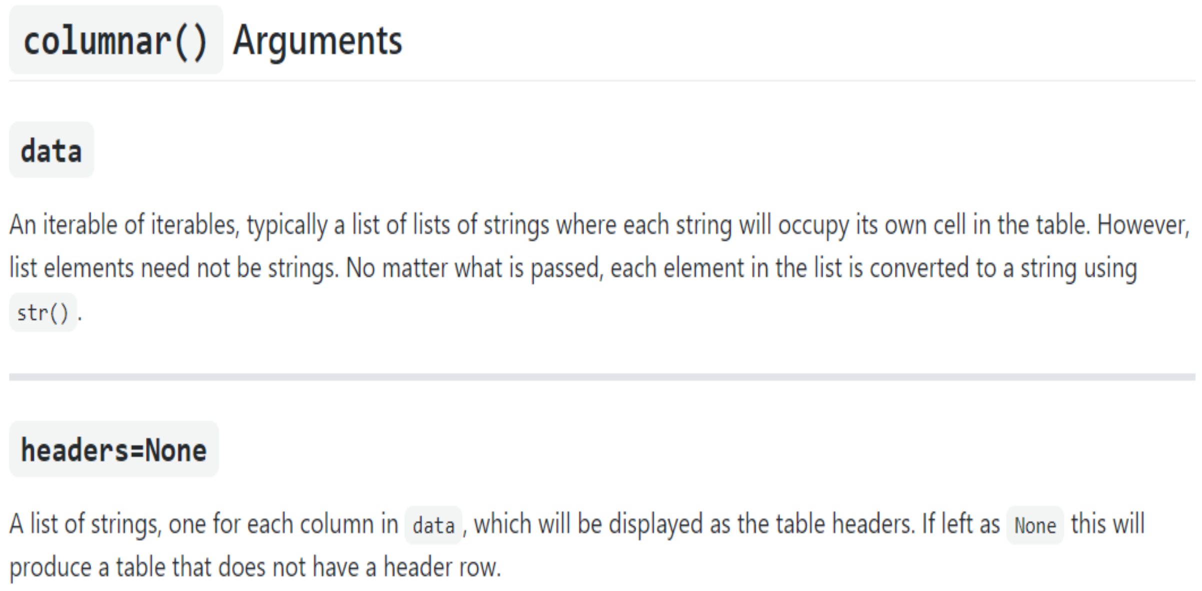
Field	Points to be remembered
subjectcode	Autogenerated. Maximum of the available subject code is selected and is incremented by 1.
subject	A subject can be added to the table only if the class does not exist in the table already.

Columnar Module

Used for displaying output in tabular form.



```
table=columnar(data,header,max_column_width=30,min_column_width=5,terminal_width=500,justify='c')
print(table)
```



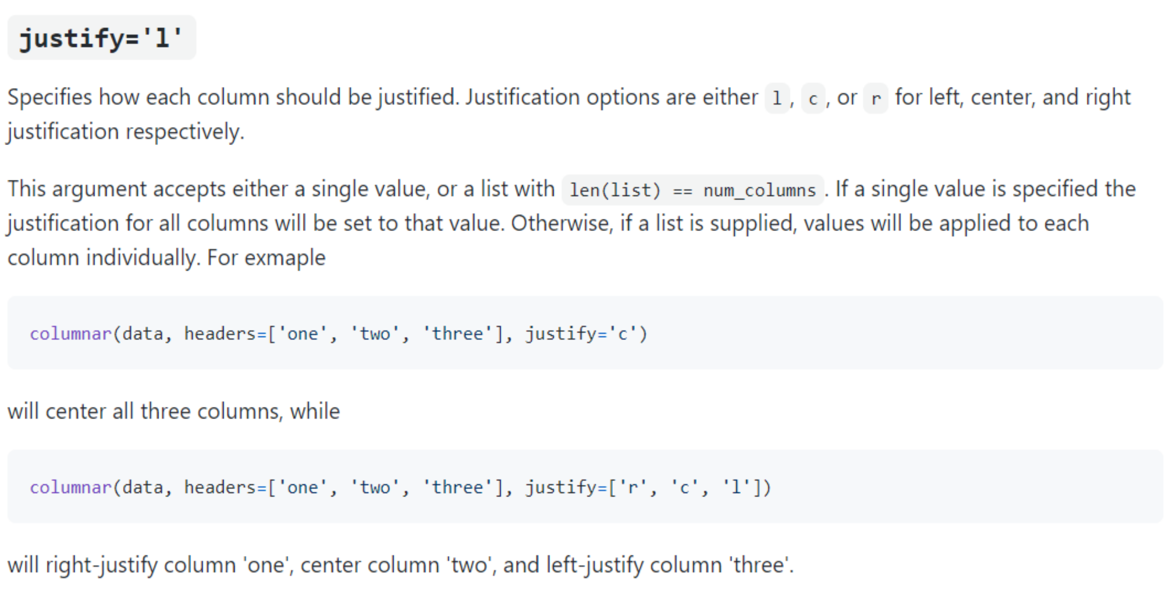
`columnar()` Arguments

`data`

An iterable of iterables, typically a list of lists of strings where each string will occupy its own cell in the table. However, list elements need not be strings. No matter what is passed, each element in the list is converted to a string using `str()`.

`headers=None`

A list of strings, one for each column in `data`, which will be displayed as the table headers. If left as `None` this will produce a table that does not have a header row.



`justify='l'`

Specifies how each column should be justified. Justification options are either `l`, `c`, or `r` for left, center, and right justification respectively.

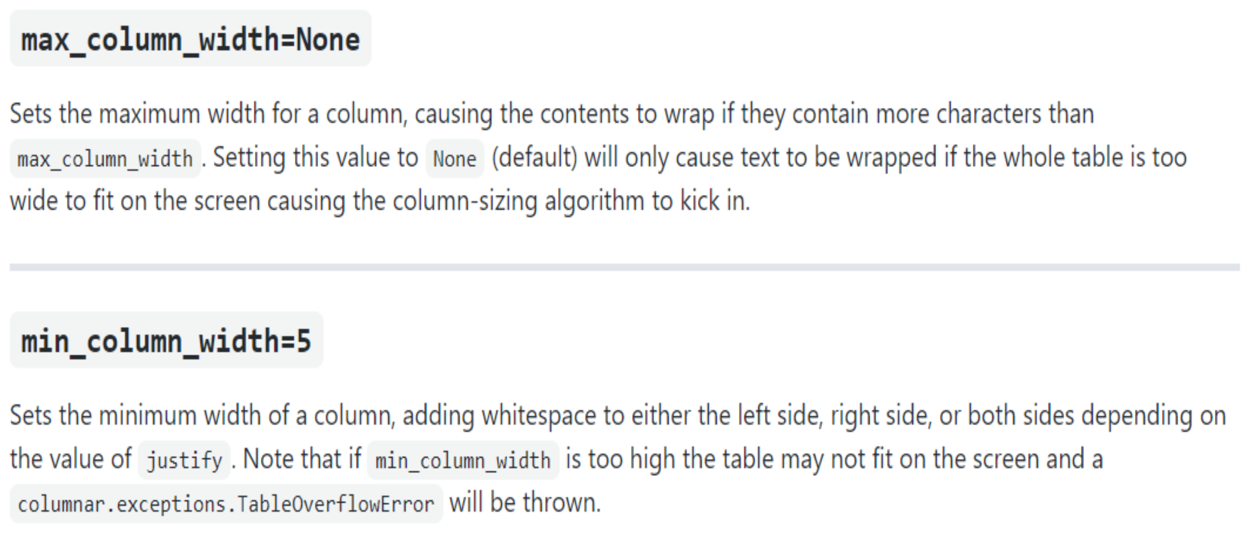
This argument accepts either a single value, or a list with `len(list) == num_columns`. If a single value is specified the justification for all columns will be set to that value. Otherwise, if a list is supplied, values will be applied to each column individually. For exmaple

```
columnar(data, headers=['one', 'two', 'three'], justify='c')
```

will center all three columns, while

```
columnar(data, headers=['one', 'two', 'three'], justify=['r', 'c', 'l'])
```

will right-justify column 'one', center column 'two', and left-justify column 'three'.

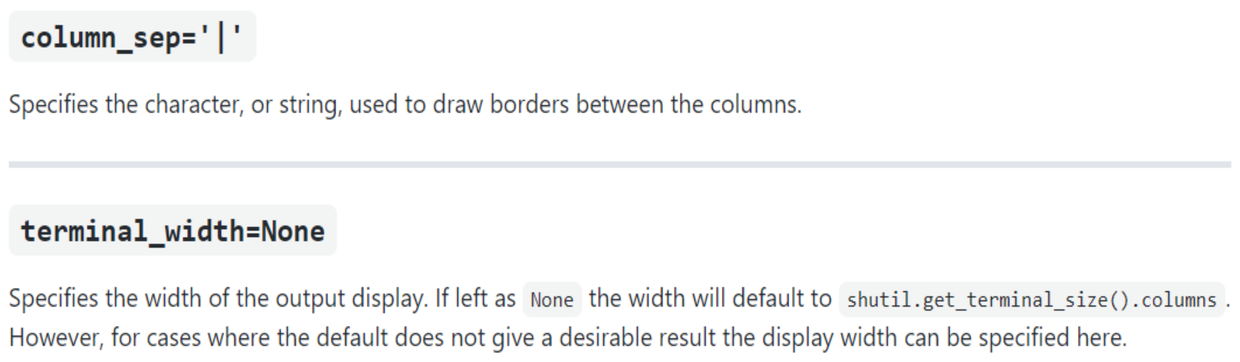


`max_column_width=None`

Sets the maximum width for a column, causing the contents to wrap if they contain more characters than `max_column_width`. Setting this value to `None` (default) will only cause text to be wrapped if the whole table is too wide to fit on the screen causing the column-sizing algorithm to kick in.

`min_column_width=5`

Sets the minimum width of a column, adding whitespace to either the left side, right side, or both sides depending on the value of `justify`. Note that if `min_column_width` is too high the table may not fit on the screen and a `columnar.exceptions.TableOverflowError` will be thrown.



`column_sep='|'`

Specifies the character, or string, used to draw borders between the columns.

`terminal_width=None`

Specifies the width of the output display. If left as `None` the width will default to `shutil.get_terminal_size().columns`. However, for cases where the default does not give a desirable result the display width can be specified here.

These are the arguments of the columnar module used in our project. For other arguments, refer this link.

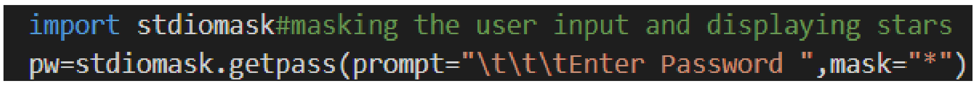
Password module:

Stdiomask.getpass is the function used for masking password input.

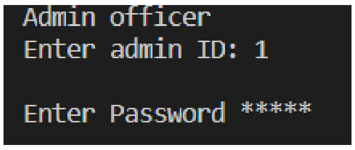
It has two arguments prompt and mask.

Prompt is the message that needs to be displayed.

Mask refers to the symbol used for masking user input.



```
import stdiomask#masking the user input and displaying stars
pw=stdiomask.getpass(prompt="\t\t\tEnter Password ",mask="*")
```



```
Admin officer
Enter admin ID: 1

Enter Password *****
```