PYTHON AND MACHINE LEARNING (BASIC)

A CERTIFICATION COURSE CONDUCTED

BY

THE SURE TRUST

Skill Upgradation for Rural-youth Empowerment

(www.suretrustforruralyouth.com)

Course Training Attended

By

BOYA DRAVEEN KUMAR

JUNE 2022 – OCTOBER 2022

Certification

This is to certify that **Mr. BOYA DRAVEEN KUMAR** has successfully completed the four months training given in "**Python and Machine Learning - Basic**" by SURE Trust during the period from June 2022 to October 2022.

Ms. Harshee Pitroda
Trainer
SURE Trust

Prof.Ch. Radha Kumari
Executive Director &
Founder, SURE Trust

Ms. Mayuri Gaikwad
Trainer
SURE Trust

Mrs. Vandana Nagesh
Director & Co-Founder, SURE Trust

TABLE OF CONTENTS

a) About the SURE TRUST

Introduction to The SURE TRUST

The SURE TRUST is born to enhance the employability of educated unemployed rural youth. It is observed that there is a wide gap between the skills acquired by students from the academic institutions and the skills required by the industry to employ them. Employability enhancement is done through giving one on one training in emerging technologies, completely through online mode. The mission of the SURE TRUST is to bridge the gap between the skills acquired and the skills required by training them in the most emerging technologies such as Artificial Intelligence (AI), Python Program, Machine Learning (ML), Deep Learning (DL), Data Science & Data Analytics, Blockchain Technology, Robotic Process Automation (RPA), and Project Management and twenty other essential and high in demand Courses ,that will enhance their employability. After completion of four months training in the course, the trainees will get live projects from industries as internship activity to get experience in applying to real time situation what they have learnt during the course. These projects will give them hands on experience which is much sought after by the prospective industry employing them. Currently students from all over India are enrolling for various courses offered by the SURE TRUST. The SURE TRUST offers every course free of cost with no financial burden of any kind to students. This initiative is purely a service-oriented one aiming to guide the rural youth who are educated but unemployed due to lack of upgradation in their skill sets. The birth of SURE TRUST is a God given boon to rural youth who could reach great heights either in employment or in entrepreneurship once they receive the training offered followed by the company internship. Many companies are coming forward to join their hands with us by offering internship projects to hand hold and lead the rural youth in their career settlement.

Vision of the SURE TRUST

The vision of the SURE TRUST is to enhance the employability of educated unemployed youth, particularly living in rural areas, through skill upgradation, with no cost to the students.

Mission of the SURE TRUST

The mission is to bridge the gap between the skills acquired in the academic institutions and the skills required in industries as a pre-condition for employment.

Functioning of the SURE TRUST

There are three dedicated, committed, and hard-working women on the board of management of the SURE TRUST who will look into the various administrative and other matters relating to the enrolment of students, organizing trainers, entering into agreements with companies for getting live projects to students as internship programs, and so on. All the three women on the board are all the alumni from Sri Sathya Sai Institute of Higher Learning, Anantapur Campus, deemed to be a university. The women board is supported by five eminent advisories who are from different walks of life and have made outstanding mark in career in their respective fields. For more details about SURE TRUST please visit the website https://suretrustforruralyouth.com/

2. Course Content

The SURE TRUST conducts a four months training for every course on a uniform basis. A session spanning across one to one & half hour is taken by the trainers for every major course. Sessions are conducted to complete the predesigned course structure within the fixed time period. Course content is designed to suit the current requirement of the industry and validated by the industry experts. The course content of all these courses is so dynamic that any changed condition noticed in the industry will automatically get reflected in the content of the respective course. As the course content is dynamic, the Following is the course content of the current course in Data Science and Data Anal Python and Machine Learning (Basic).

Python & Machine Learning – BASIC Course

Objective:

To train the learners to program in Python, analyzing and visualizing data, developing & evaluating models; and get introduced to complex computations.

Course Content:

1. Python for data science and machine learning

- Python basics
- Python OOPs

2. Data analysis

- Pandas library
- Numpy library

3. Data visualization

- Matplotlib
- Seaborn

4. Exploratory data analysis

- Univariate
- Bivariate
- Derived metrics
- Introduction to databases (MySQL for Data science students)

5. Math for machine learning

- Statistics
- Linear algebra
- Calculus
- Probability theory

6. Supervised learning

- Feature selection
- Regression
- Linear regression
- Polynomial regression

- Advanced regression introduction
- Classification
- Logistic regression
- Naive Bayes
- Support vector machine
- Tree models

7. Advanced regression

- Regularized regression
- Ridge and Lasso regression
- Model selection and Grid Search methods

8. Unsupervised learning

- K means clustering
- Hierarchical clustering
- Principal component analysis

9. Theory introduction to deep learning and reinforcement learning

- Theory about ANN, CNN, RNN, LSTM
- Theory about Markov Decision process

3.Conduct of the course:

a) Modalities for the conduct of all the courses are fixed by the SURE TRUST which are uniformly followed across the courses.

- Mode of Training --- Online
- Period of Training --- Four months
- Sessions per week --- 3 to 6
- Length of the session --- 1 to 2 hours
- Tests to be taken --- 2 per month
- Assignments --- 2 per month
- Last 15 days --- Final practice and preparing the course report

b) Student byelaws:

Students enrolling for the courses under SURE TRUST are strictly required to follow the following byelaws set for them.

Byelaws for students to become eligible for certificate at the end of the course

I. Minimum Attendance:

Every student must put in a minimum of 85% attendance in attending the classes for getting the eligibility to receive the certificates.

II. Two written tests are to be taken in each month:

Since the objective of the certification program is to turn out well qualified students from the respective courses, minimum two written tests are to be taken in each month for each course to ensure that the students are pulled along the expected line of standard.

III. Assignment submissions:

Ten exercises constituting one assignment for every two to three new functions/topics taught, resulting in minimum seven such assignments are to be submitted during the four months period.

IV. Preparing the final course report in the prescribed format:

During the last fifteen days in the fourth month, students many be asked to consolidate and compile all the assignments submitted in a word document along with the other chapters which will constitute a course report for each student. This report will be the unique contribution a student carries from the trust to show case the rigorous training he/she received during the four months period. Besides the report will stand as a testimony for the detailed learning a student has acquired in the chosen area. This will facilitate the industry in handpicking the required student for the job.

V. External Viva-voce:

Every student has to successfully clear the external viva-voce arranged in their respective course.

VI. KYC norms:

Each student wishing to enroll for the course must submit a written letter saying that he/she will not drop from the course until its completion, which will also be signed by father / mother besides the student himself / herself.

VII. Attend the full class:

All the students are expected to attend each class for full duration. Some students are observed moving out of classes after logging in which does not go well with the learning objective of students.

VIII. Ensure discipline in the group:

All the students are advised strictly to follow group etiquette and restrain from posting in the group any unethical messages or teasing messages or personal interactive messages. This group is purely created for academic purpose and hence only academic interactions should go on

Python and Machine Learning Basic

The course is taught to the students as per the syllabus prescribed and as per the course modalities keeping in mind the bylaws set for them. Periodical tests are conducted to assess the understanding of the students in the course. Assignments are given to ensure that students gain versality in the python and machine learning concepts. The assignments of the student are given below. These assignments which are done with high creativity and out of box thinking not only reflect the student's innovativeness but also constitute solved exercises in the Various Data set for the fresh learners to practice and gain confidence. The assignments are listed below in the order of their conducting during the entire course.

PYTHON ASSIGNMENTS:

Assignment-1:

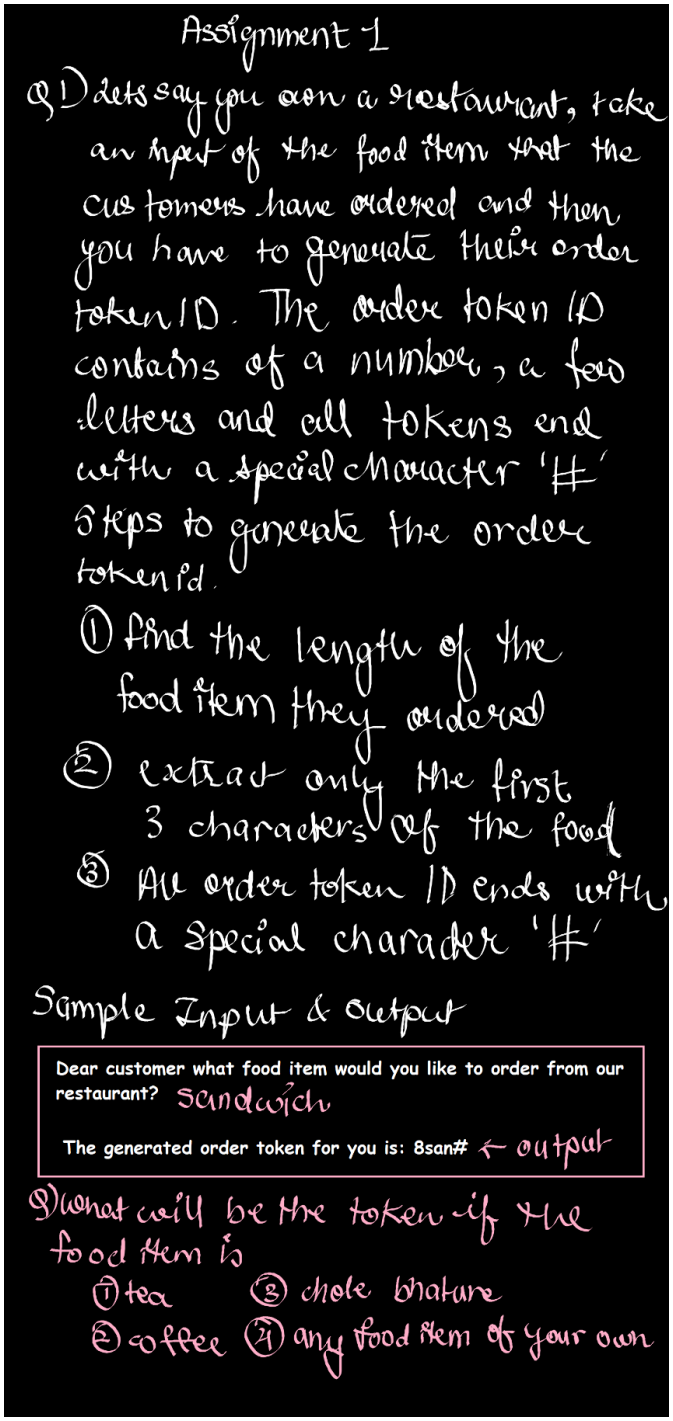
Name: BOYA DRAVEEN KUMAR

Trainer Name: HARSHEE PITRODA

Batch Number: G-15

Course Name: Machine learning With Python

Date:25-May-2022

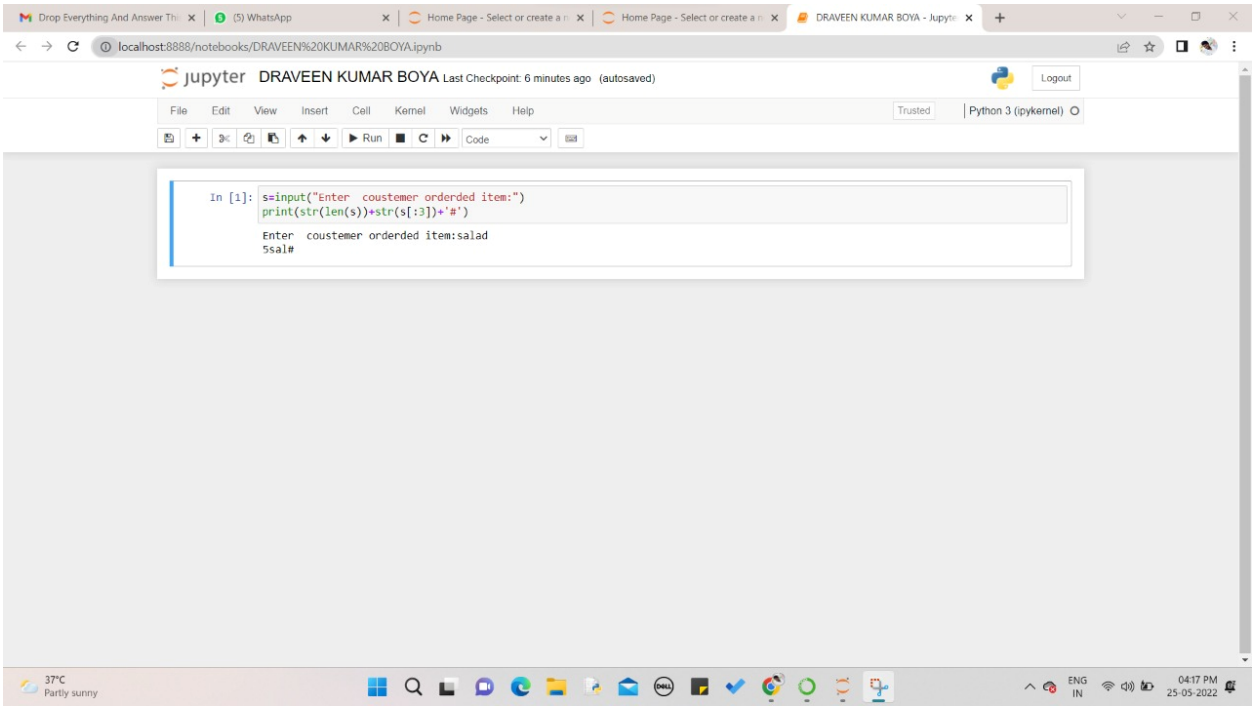


Code1:

#QUESTION 1

S=input("Enter coustemer orderded item:")

Print(str(len(s))+str(s[:3])+’#’)



Sample Input:salad

Sample Output:5sal#

Assignment 1

Q2) You work at XYZ college and its the 'admission season'! You are one of the IT support at the XYZ college. You need to generate admission token ID for them based on the following things -

1. their first name
2. their 10th standard percentage (only in numbers)
3. their marks in maths, science & english subjects
4. Start & end all token ID with XYZ choice.

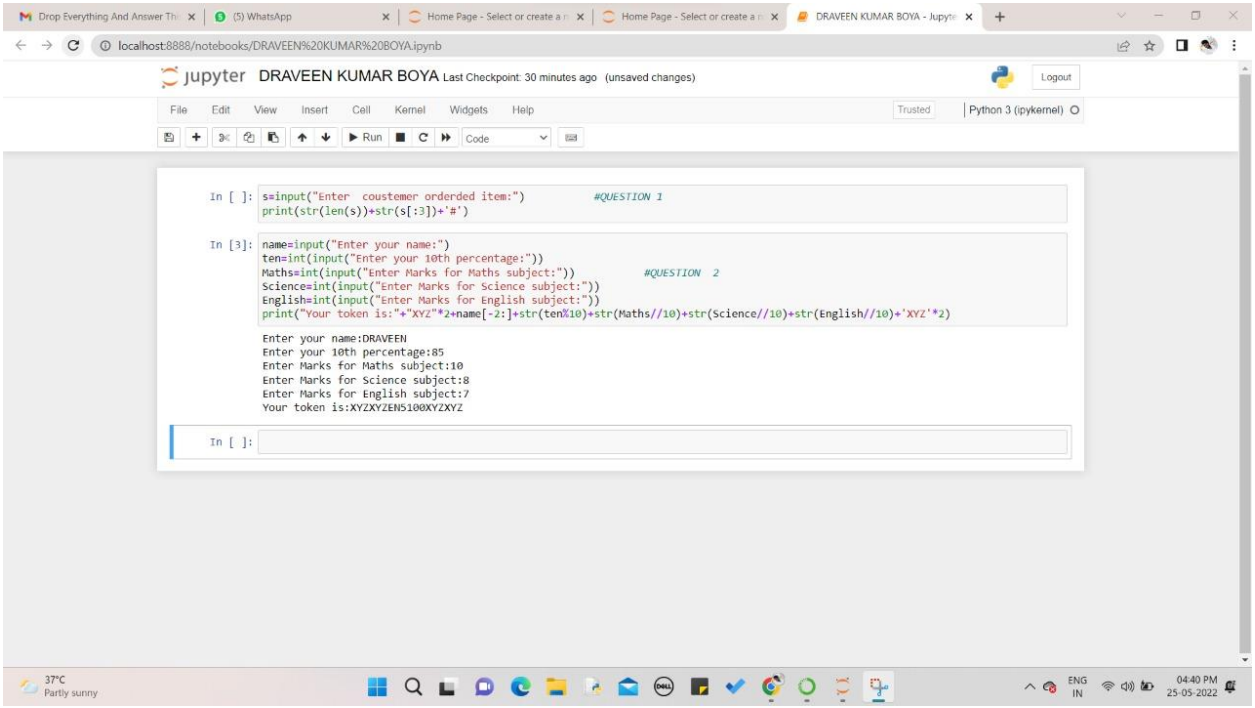
To generate the token ID follow these steps

1. take input of their name & extract last 2 characters of their first name
2. take input of their 10th %, & extract only the last digit
3. take input of their marks for math, science & english subject & extract only the first digit
4. this ID should start & end with 'XYZ' choice.

Enter your name: Raashi
Enter your 10th Percentage: 81
Enter marks for maths subject: 74
Enter marks for science subject: 82
Enter marks for english subject: 75
Your token is: XYZXYZhi1787XYZXYZ

Code:

```
#QUESTION  2
Name=input("Enter your name:")
Ten=int(input("Enter your 10th percentage:"))
Maths=int(input("Enter Marks for Maths subject:"))
Science=int(input("Enter Marks for Science subject:"))
English=int(input("Enter Marks for English subject:"))
Print("Your token is:"+"XYZ"*2+name[2:]+str(ten%10)+str(Maths//10)+str(Science//10)+str(English//10)+'XYZ'*2)
```



Sample Input:

Enter your name:DRAVEEN

Enter your 10th percentage:85

Enter Marks for Maths subject:10

Enter Marks for Science subject:8

Enter Marks for English subject:7

Sample Output:

Your token is:XYZXYZEN5100XYZXYZ

----------xxxxxx----------

Assignment-2:

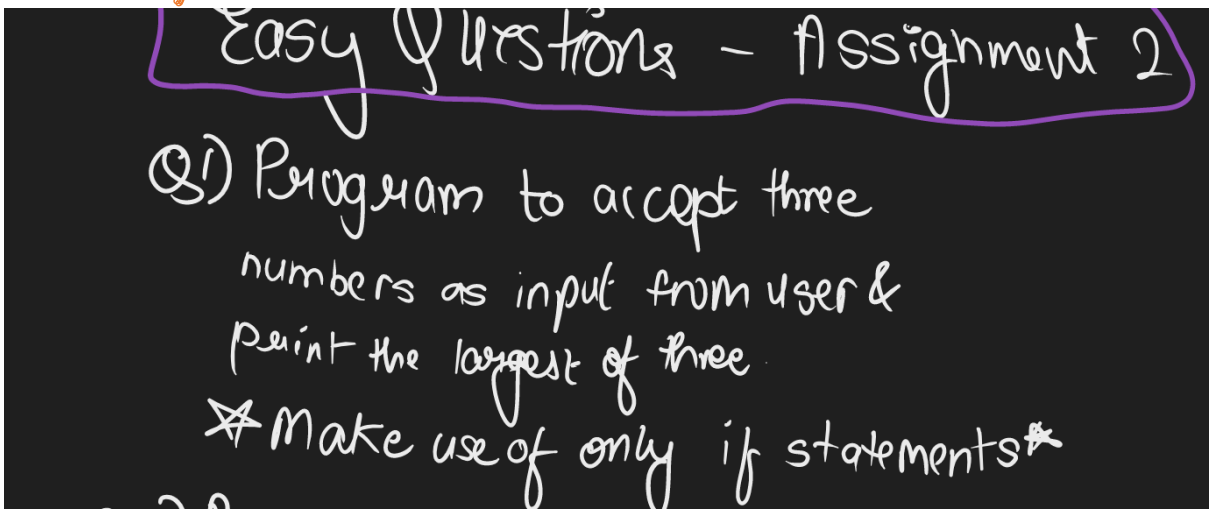
Name:BOYA DRAVEEN KUMAR

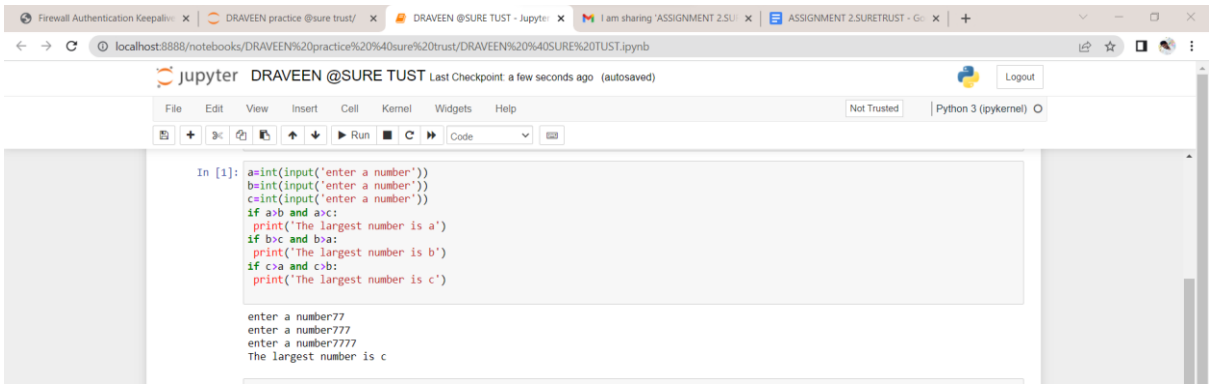
Trainer Name:HARSHEE PITRODA

Batch Number: G-15

Course Name: Mechainlearning With Python

Date:31-May-2022



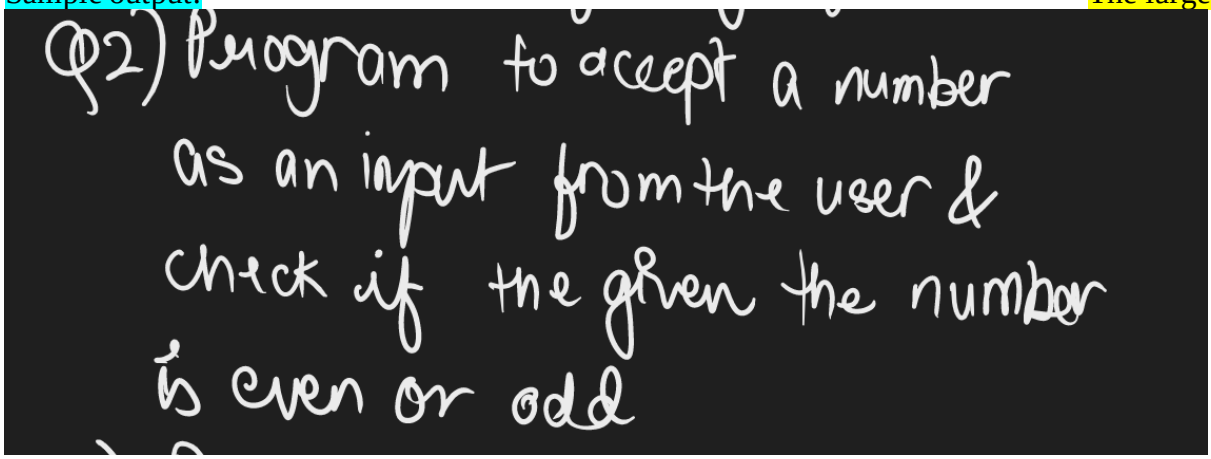


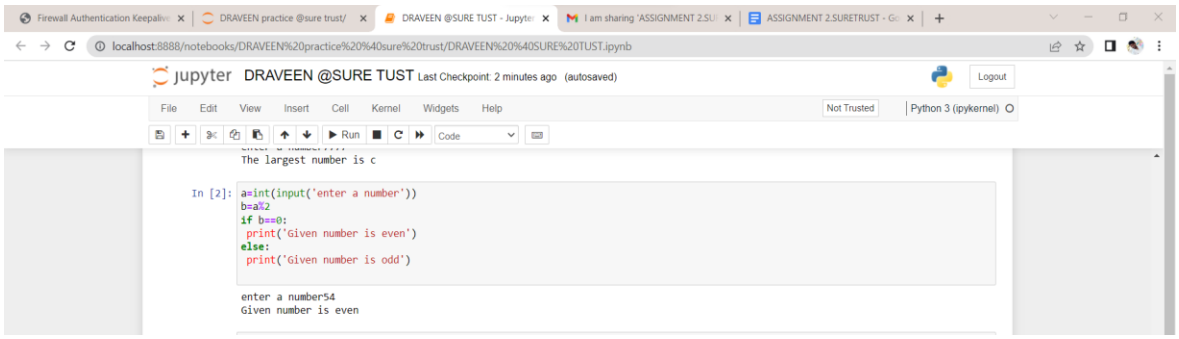
a=int(input('enter a number'))

b=int(input('enter a number'))

```
c=int(input('enter a number'))
if a>b and a>c:
 print('The largest number is a')
if b>c and b>a:
 print('The largest number is b')
if c>a and c>b:
 print('The largest number is c')
```

Sample input: enter a numbe
r77
enter a number777
enter a number7777
Sample output: The largest number is c






```
a=int(input('enter a number'))
b=a%2
if b==0:
 print('Given number is even')
```

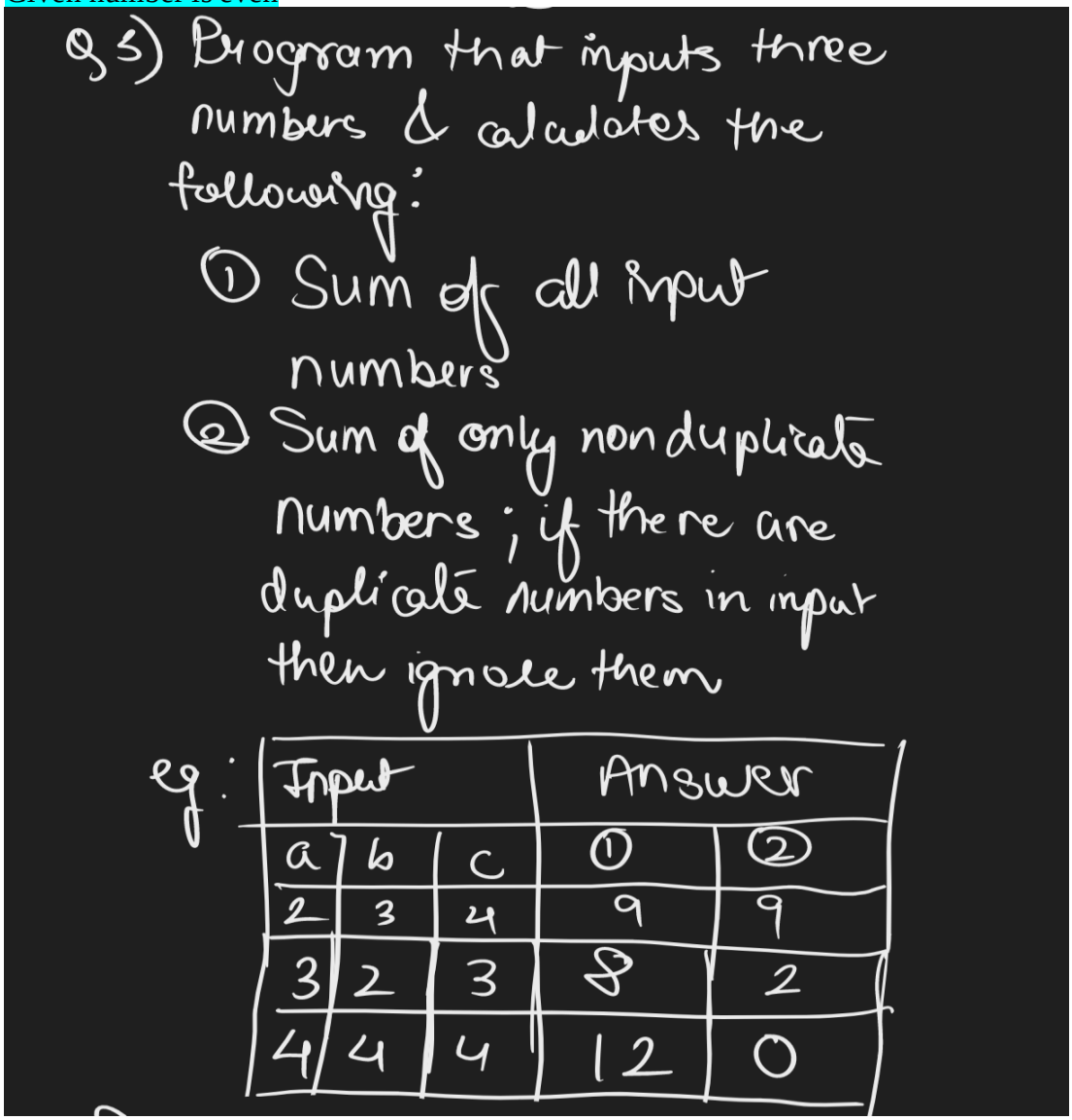
else:

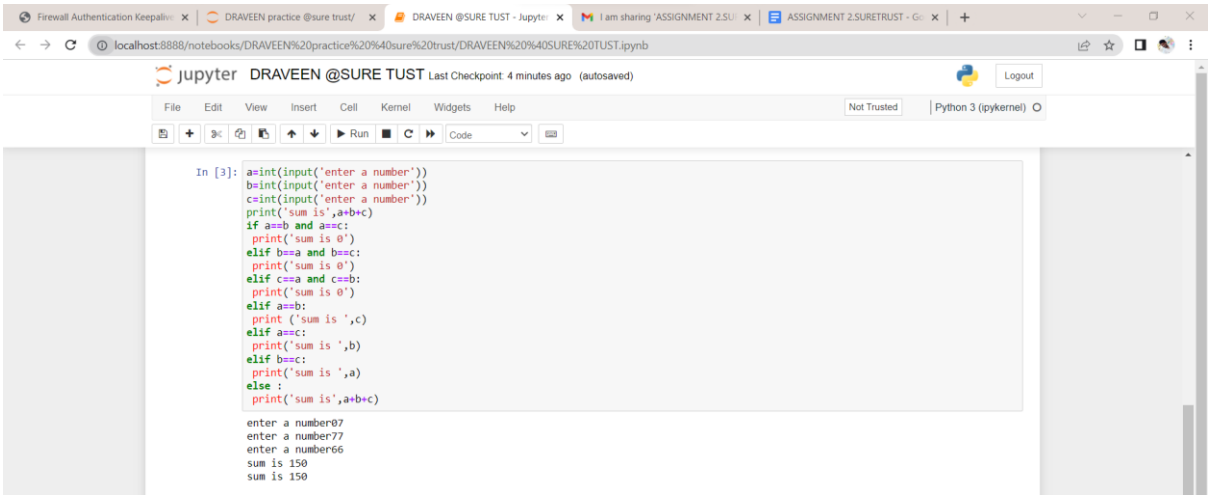
print('Given number is odd')

Smaple input:

enter a number54

Sample output:
Given number is even






```
a=int(input('enter a number'))
b=int(input('enter a number'))
c=int(input('enter a number'))
print('sum is',a+b+c)
if a==b and a==c:
 print('sum is 0')
elif b==a and b==c:
 print('sum is 0')
elif c==a and c==b:
 print('sum is 0')
elif a==b:
 print ('sum is ',c)
elif a==c:
 print('sum is ',b)
elif b==c:
 print('sum is ',a)
else :
 print('sum is',a+b+c)
```

sample input:

enter a number07

enter a number77

enter a number66

Sample output:

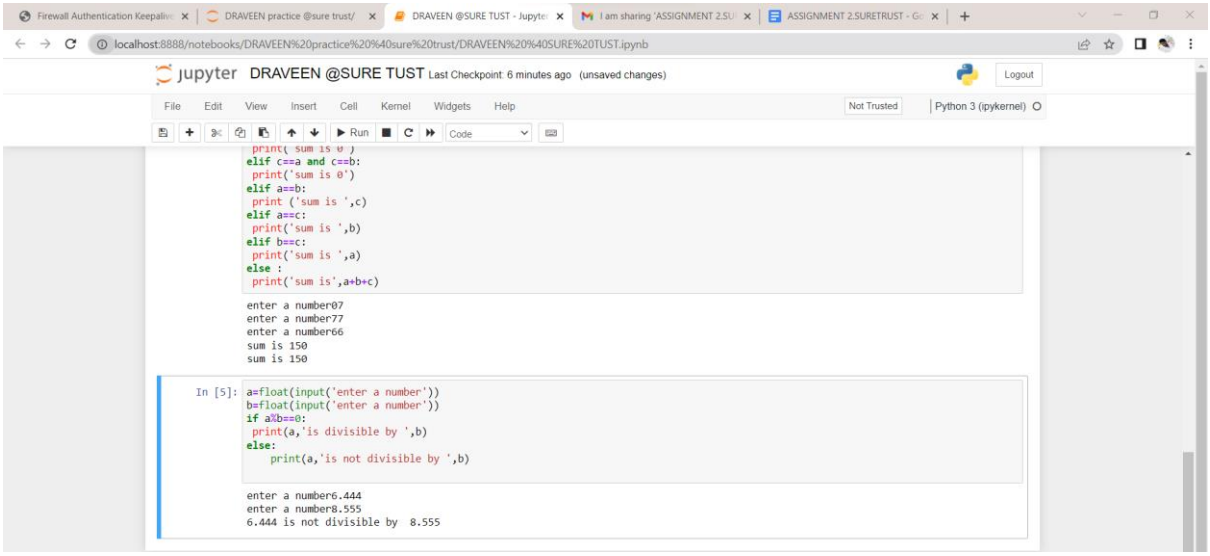
sum is 150

sum is 150

Q4) Program to test divisibility a number from another number

eg:

```
Enter first number : 119
Enter second number : 3
119.0 is not divisible by 3.0
====================
Enter first number : 119
Enter second number : 17
119.0 is divisible by 17.0
======================
Enter first number : 1234.30
Enter second number : 5.5
1234.3 is not divisible by 5.5
```



a=float(input('enter a number'))

b=float(input('enter a number'))

```
if a%b==0:
 print(a,'is divisible by ',b)
else:
    print(a,'is not divisible by ',b)
```

Sample input:

```
enter a number6.444
enter a number8.555
```

Sample output:

```
6.444 is not divisible by  8.555
```

----------xxxxxxx----------

Assignment-3:

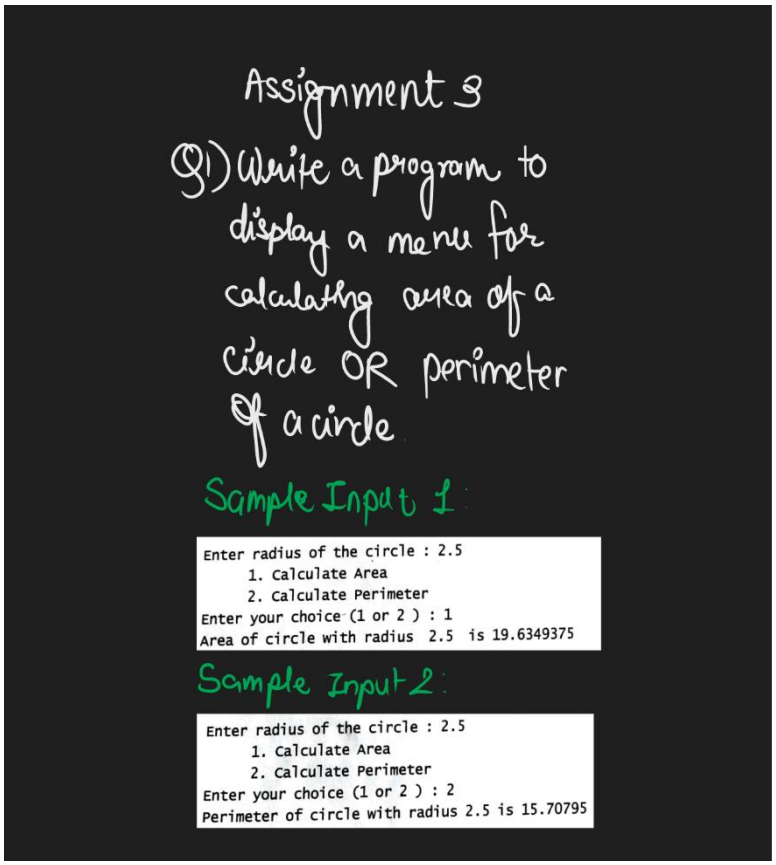
Name: BOYA DRAVEEN KUMAR

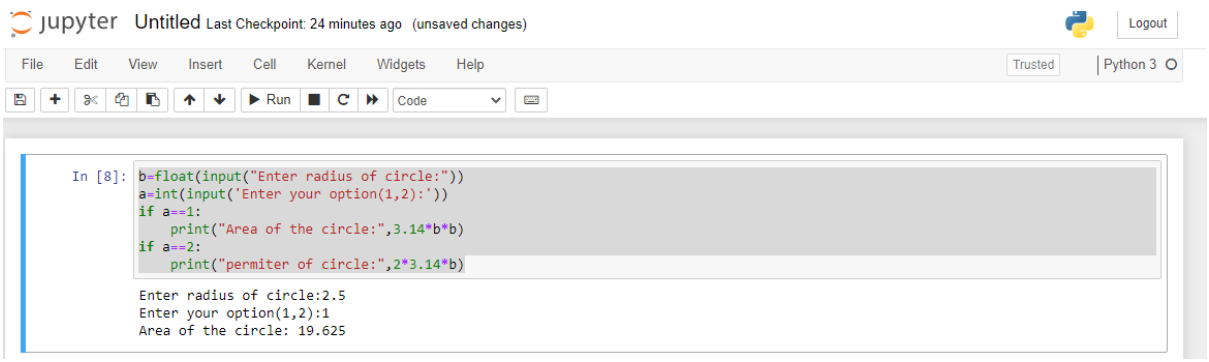
Trainer Name:HARSHEE PITRODA

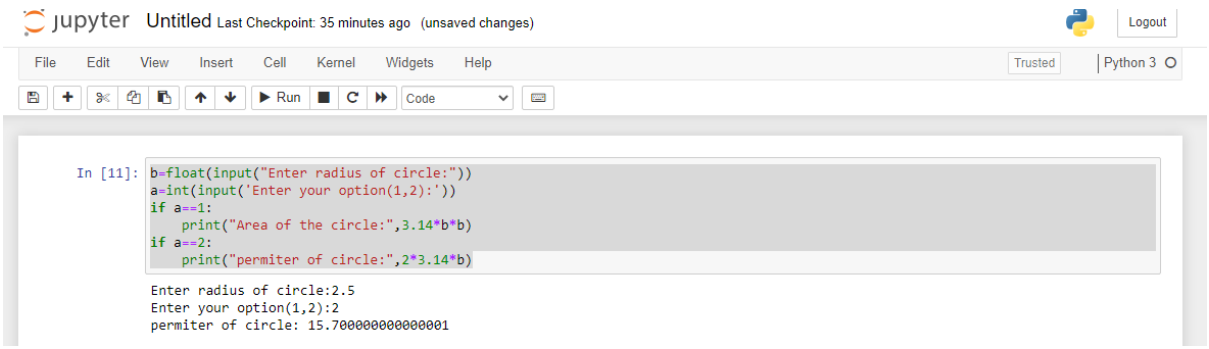
Batch Number: 09(G-15)

Course Name: Mechain learning with python

Date:02-Jun-2022







b=float(input("Enter radius of circle:"))

a=int(input('Enter your option(1,2):'))

```
if a==1:
    print("Area of the circle:",3.14*b*b)
if a==2:
    print("permiter of circle:",2*3.14*b)
```

Sample input:

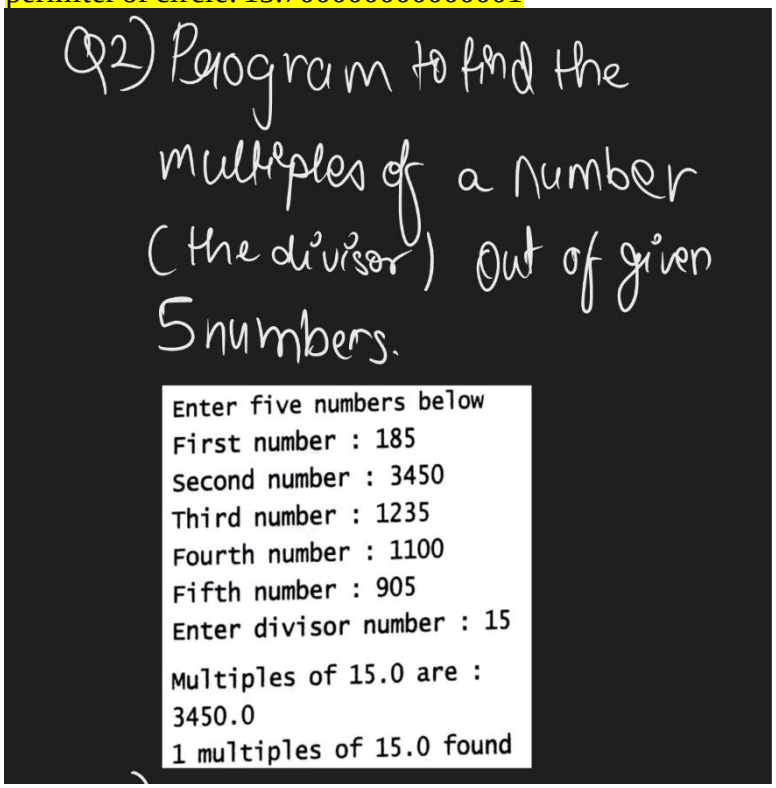
Enter radius of circle:2.5
Enter your option(1,2):1

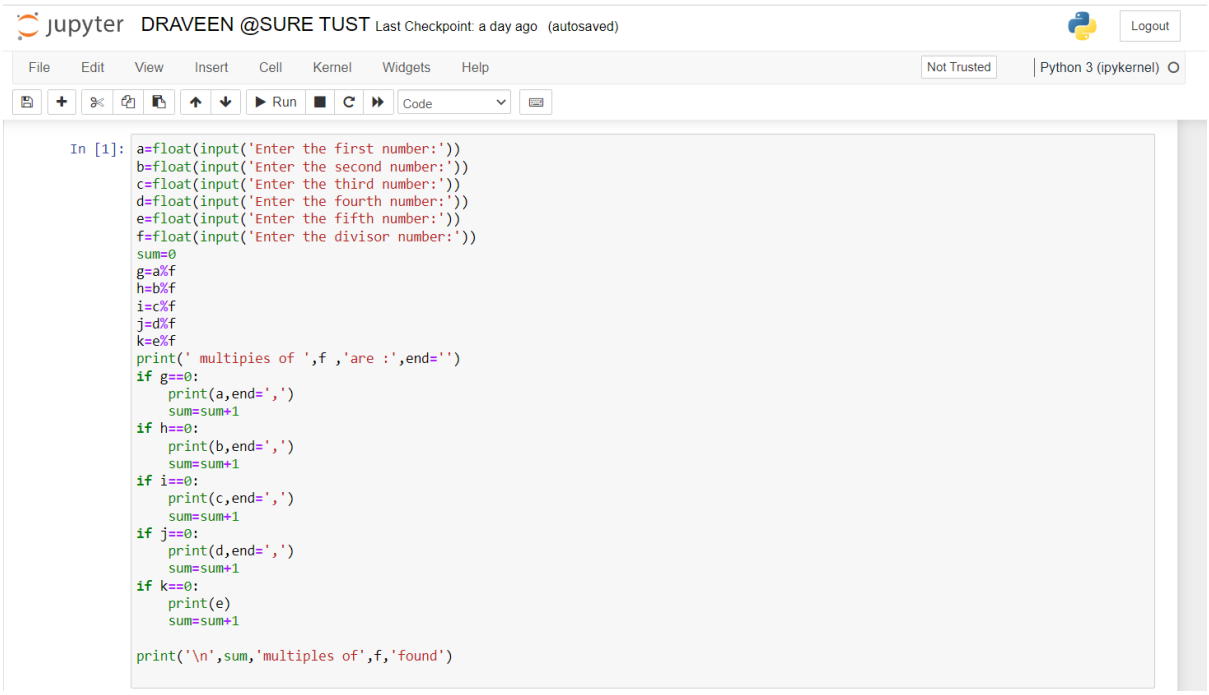
Sample output:
Area of the circle: 19.625

Sample input:

Enter radius of circle:2.5
Enter your option(1,2):2

Sample output:
permiter of circle: 15.700000000000001






```
Enter the first number:185
Enter the second number:3450
Enter the third number:1235
Enter the fourth number:1100
Enter the fifth number:905
Enter the divisor number:15
 multipies of  15.0 are :3450.0,
 1 multiples of 15.0 found
```

a=float(input('Enter the first number:'))

b=float(input('Enter the second number:'))

c=float(input('Enter the third number:'))

d=float(input('Enter the fourth number:'))

e=float(input('Enter the fifth number:'))

f=float(input('Enter the divisor number:'))

sum=0

g=a%f

h=b%f

i=c%f

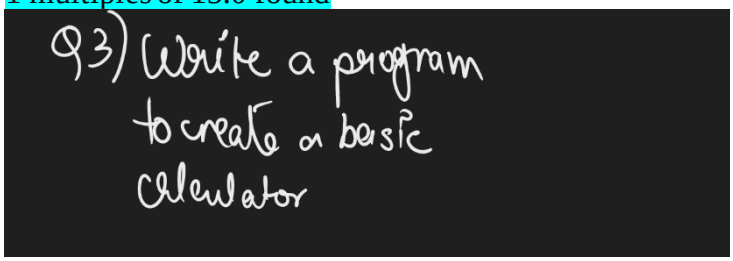
```
j=d%f
k=e%f
print(' multipies of ',f ,'are :',end='')
if g==0:
    print(a,end=',')
    sum=sum+1
if h==0:
    print(b,end=',')
    sum=sum+1
if i==0:
    print(c,end=',')
    sum=sum+1
if j==0:
    print(d,end=',')
    sum=sum+1
if k==0:
    print(e)
    sum=sum+1

print('\n',sum,'multiples of',f,'found')
```

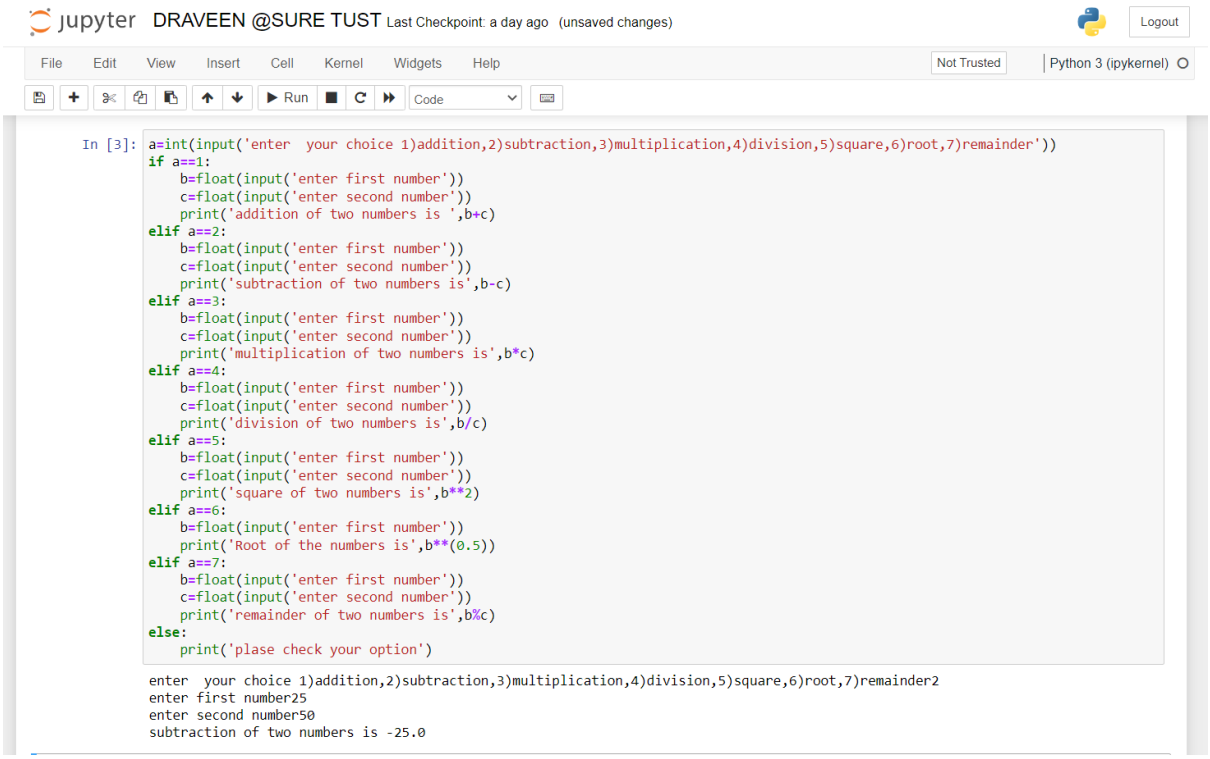
Sample input:

```
Enter the first number:185
Enter the second number:3450
Enter the third number:1235
Enter the fourth number:1100
Enter the fifth number:905
Enter the divisor number:15
 multipies of  15.0 are :3450.0,
```

Sample output:

1 multiples of 15.0 found






```
a=int(input('enter your choice 1)addition,2)subtraction,3)multiplication,4)division,5)square,6)root,7)remainder'))
if a==1:
    b=float(input('enter first number'))
    c=float(input('enter second number'))
    print('addition of two numbers is ',b+c)
elif a==2:
    b=float(input('enter first number'))
```

```
    c=float(input('enter second number'))
    print('subtraction of two numbers is',b-c)
elif a==3:
    b=float(input('enter first number'))
    c=float(input('enter second number'))
    print('multiplication of two numbers is',b*c)
elif a==4:
    b=float(input('enter first number'))
    c=float(input('enter second number'))
    print('division of two numbers is',b/c)
elif a==5:
    b=float(input('enter first number'))
    c=float(input('enter second number'))
    print('square of two numbers is',b**2)
elif a==6:
    b=float(input('enter first number'))
    print('Root of the numbers is',b**(0.5))
elif a==7:
    b=float(input('enter first number'))
    c=float(input('enter second number'))
    print('remainder of two numbers is',b%c)
else:
    print('plase check your option')
```

Sample input:

```
enter  your choice 1)addition,2)subtraction,3)multiplication,4)division,5)square,6)root,7)remainder2
enter first number25
enter second number50
```

Sample output:

subtraction of two numbers is -25.0

----------XXXXXX----------

Assignment-4:

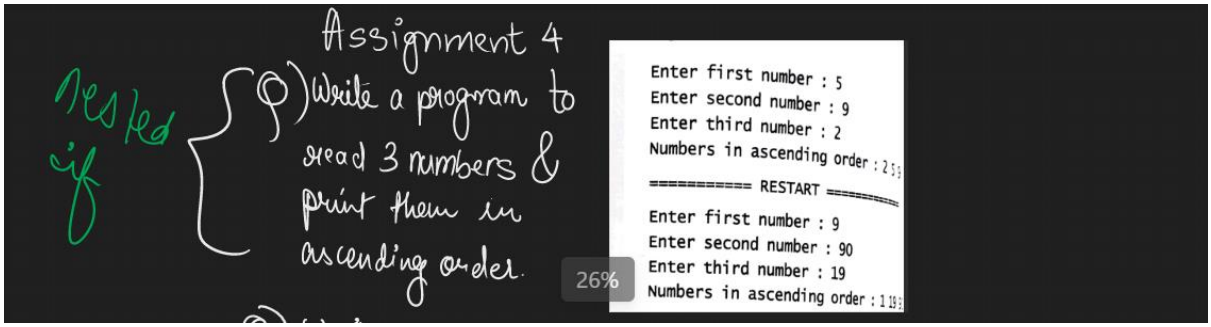
Name: BOYA DRAVEEN KUMAR

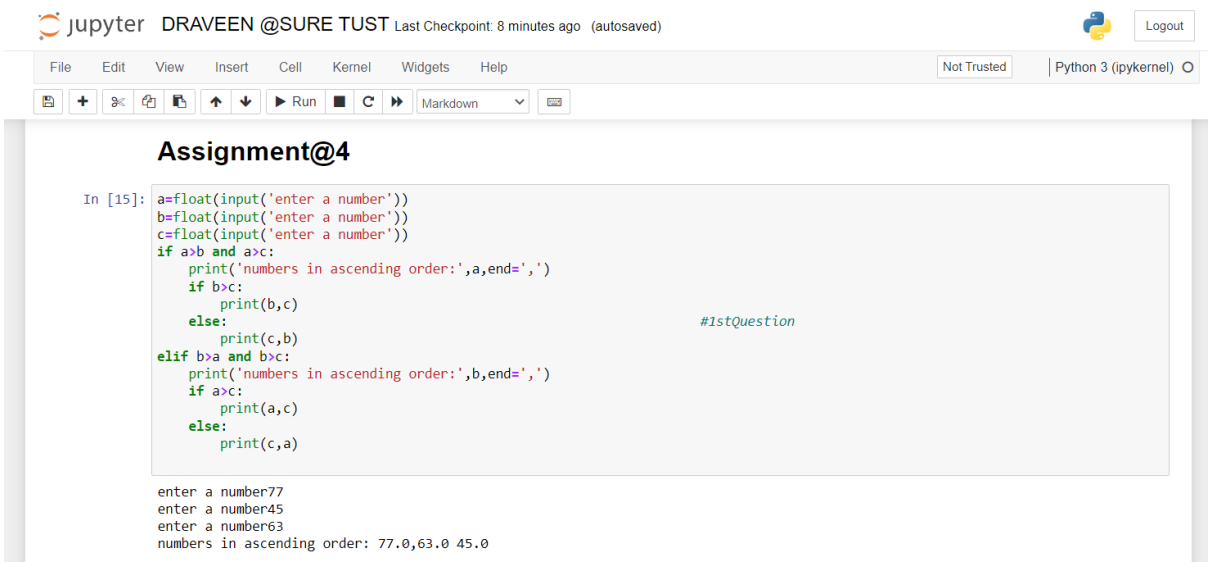
Trainer Name: HARSHEE PITRODA

Batch Number: 09(G-15)

Course Name: Mechain learning with python

Date: 04-Jun-2022





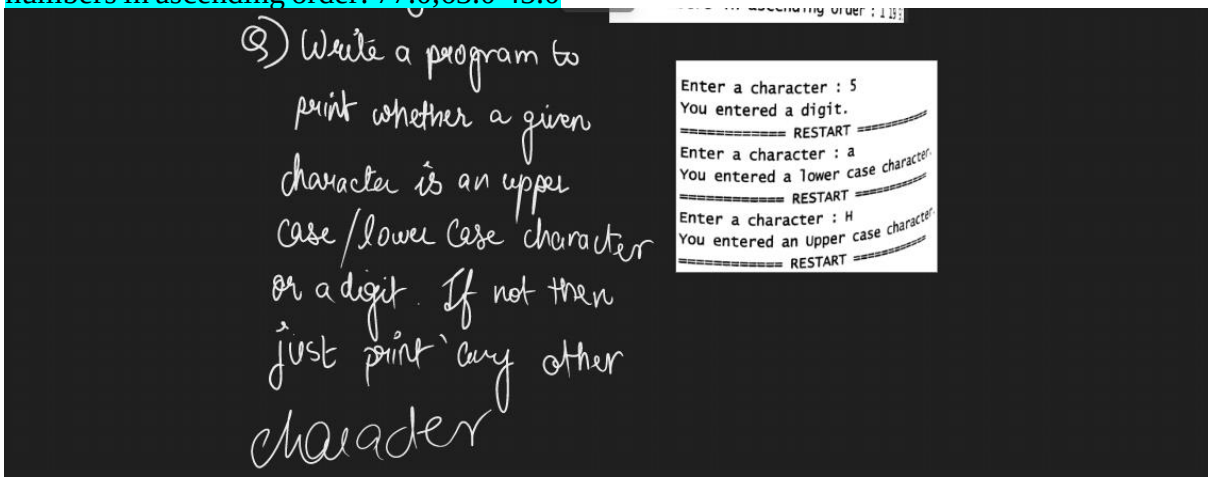
a=float(input('enter a number'))

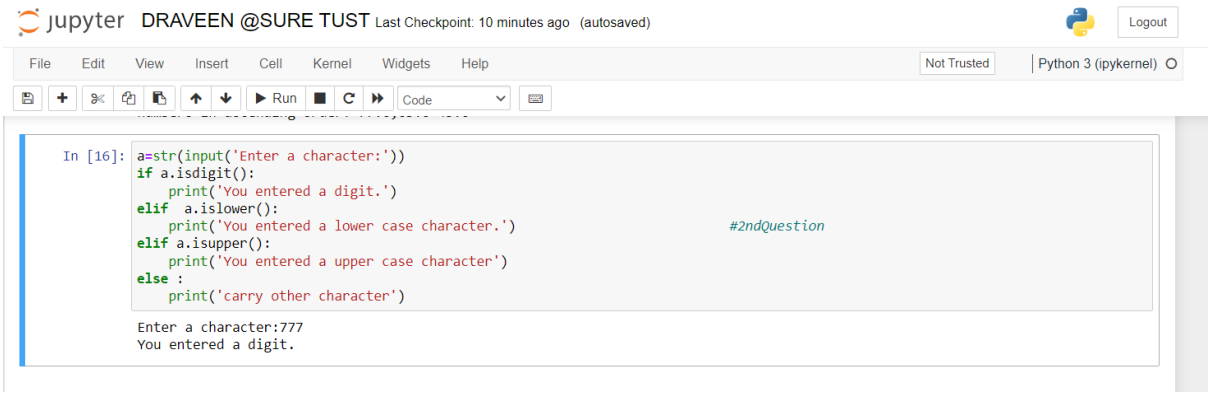
```
b=float(input('enter a number'))
c=float(input('enter a number'))
if a>b and a>c:
    print('numbers in ascending order:',a,end=',')
    if b>c:
        print(b,c)
    else:
        print(c,b)
elif b>a and b>c:
    print('numbers in ascending order:',b,end=',')
    if a>c:
        print(a,c)
    else:
        print(c,a)
```

Sample input:

enter a number77

enter a number45
enter a number63
Sample output:
numbers in ascending order: 77.0,63.0 45.0






```
a=str(input('Enter a character:'))
if a.isdigit():
    print('You entered a digit.')
elif  a.islower():
    print('You entered a lower case character.')
elif a.isupper():
    print('You entered a upper case character')
else :
    print('carry other character')
```

Sample input:

Enter a character:777

Sample output:

You entered a digit.

----------XXXXXX----------

Assignment-5:

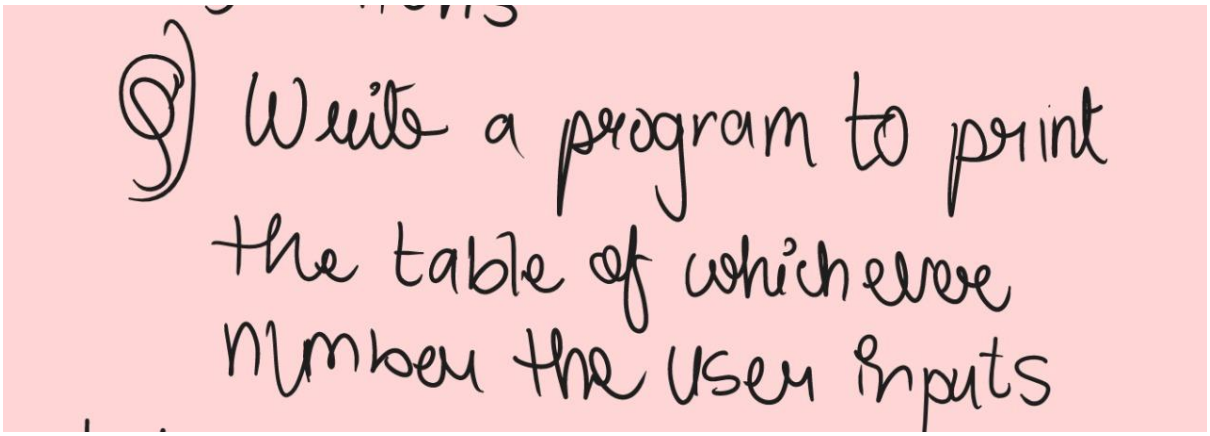
Name: BOYA DRAVEEN KUMAR

Trainer Name:HARSHEE PITRODA

Batch Number: 09(G-15)

Course Name: Mechain learning with python

Date:06-Jun-2022



jupyter DRAVEEN @SURE TUST Last Checkpoint: 19 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

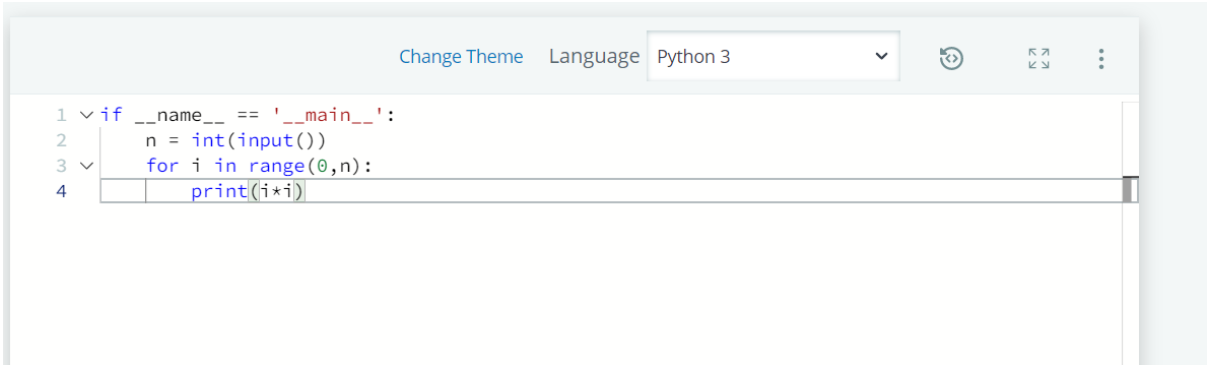
In [8]:
```
if __name__ == '__main__':
    n = int(input())
    for i in range(0,n):
        print(i*i)
```

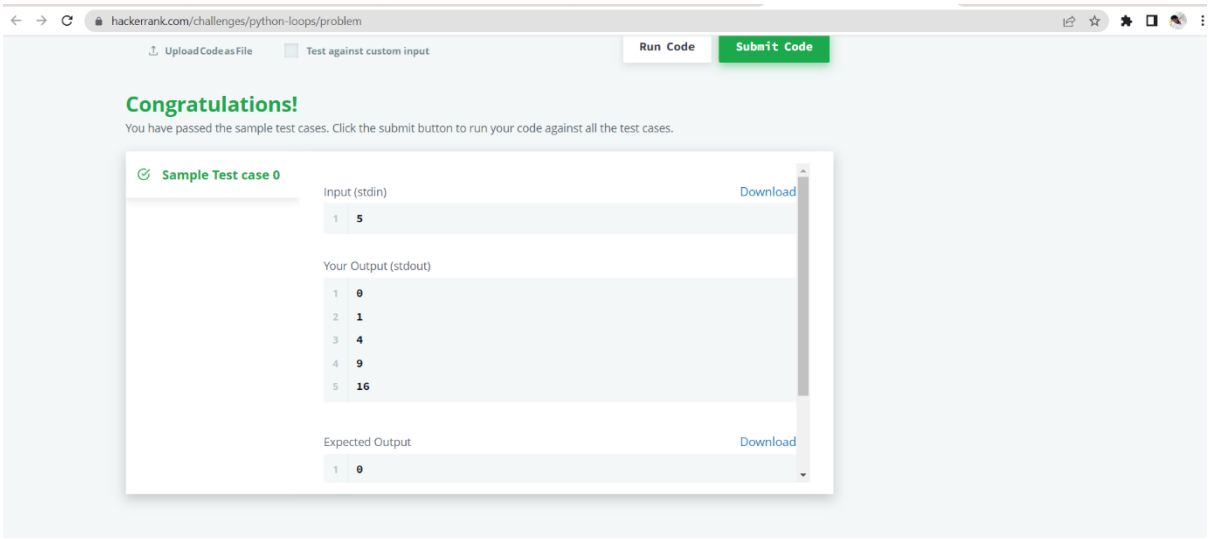
```
5
0
1
4
9
16
```

In [1]:
```
num=int(input("Enter the number for which you need the Table"))
for i in range(1,11):
    print("{}*{}={}".format(num,i,num*i))
```

```
Enter the number for which you need the Table7
7*1=7
7*2=14
7*3=21
7*4=28
7*5=35
7*6=42
7*7=49
7*8=56
7*9=63
7*10=70
```

https://www.hackerrank.com/challenges/python-loops/problem





https://www.hackerrank.com/challenges/py-if-else/problem?isFullScreen=true

Change Theme Language Python 3

```python
#!/bin/python3

import math
import os
import random
import re
import sys
if __name__ == '__main__':
    n = int(input().strip())
if(n%2!=0):
    print("Weird")
else:
    if(n>=2 and n>=5):
        print("Not Weird")
    elif(n>=6 and n>=20):
        print("Weird")
    else:
        print("Not Weird")

```

Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Sample Test case 1

Input (stdin) Download

```
3
```

Your Output (stdout)

```
Weird
```

Expected Output Download

```
Weird
```

https://www.hackerrank.com/challenges/whats-your-name/problem?isFullScreen=true

Change Theme Language Python 3

```python
#
# Complete the 'print_full_name' function below.
#
# The function is expected to return a STRING.
# The function accepts following parameters:
#  1. STRING first
#  2. STRING last
#

def print_full_name(first, last):
    # Write your code here
    first_name=first
    last_name=last
    a="Hello {} {}! You just delved into python."
    print(a.format(first_name,last_name))
    return None
if __name__ == '__main__':
```

Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin) Download

```
Ross
Taylor
```

Your Output (stdout)

```
Hello Ross Taylor! You just delved into python.
```

Expected Output Download

```
Hello Ross Taylor! You just delved into python.
```

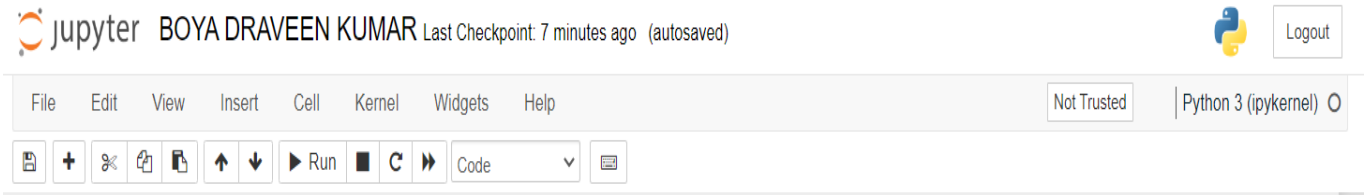
----------XXXXXX----------

MINI PROJECT -1

Date:06-07-2022
Name of the Student: BOYA DRAVEEN KUMAR
Name of the trainer: HARSHEE PITRODA
Course: Machine Learning With Python

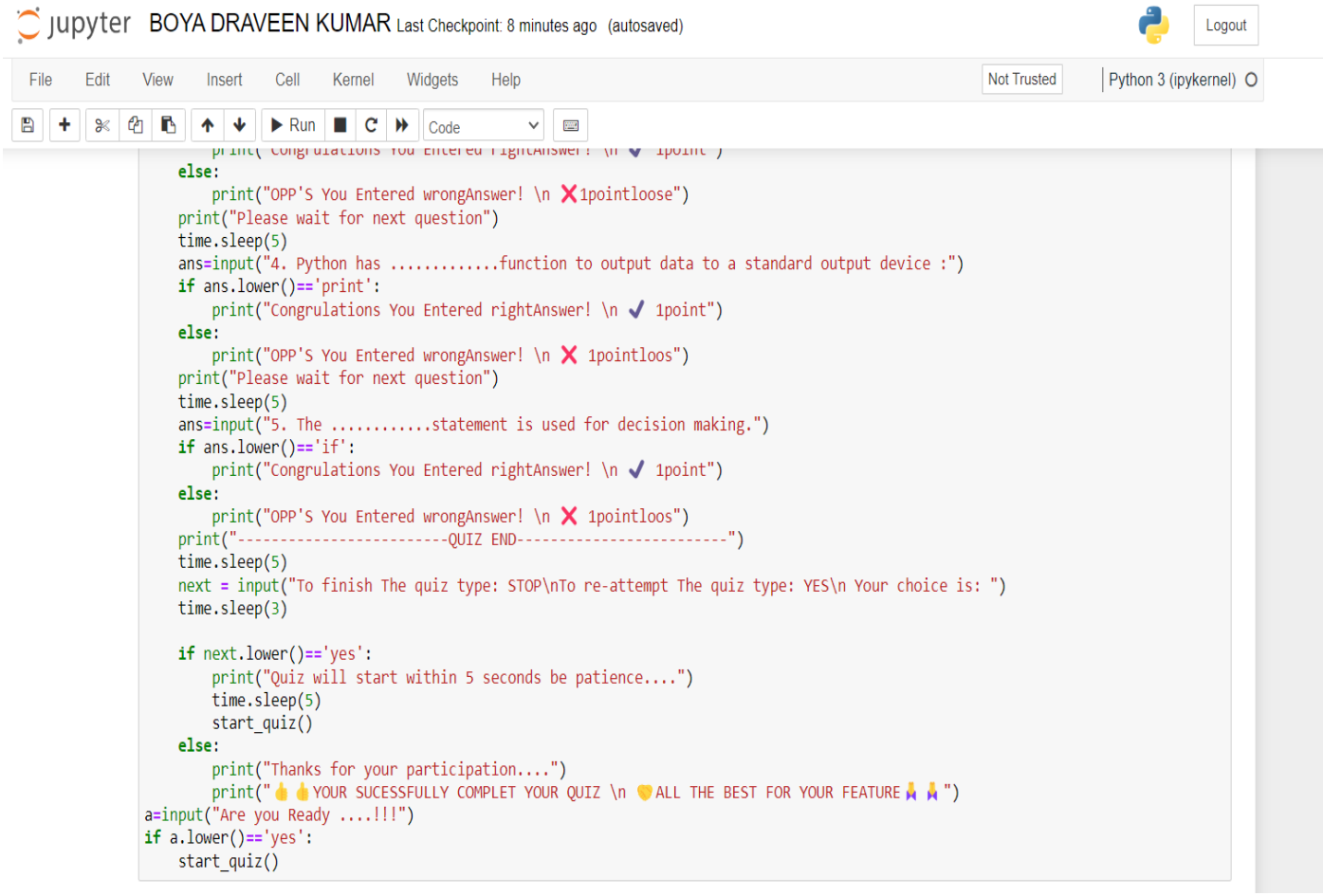
Q) You need to create a quiz game in python. The quiz needs to contain 5 questions relating to the subject python. The Quiz game asks the player questions about python. Each correct answer will score a point. At the end of the game, the program will reveal the player's final score.
You have all freedom to create the quiz game –

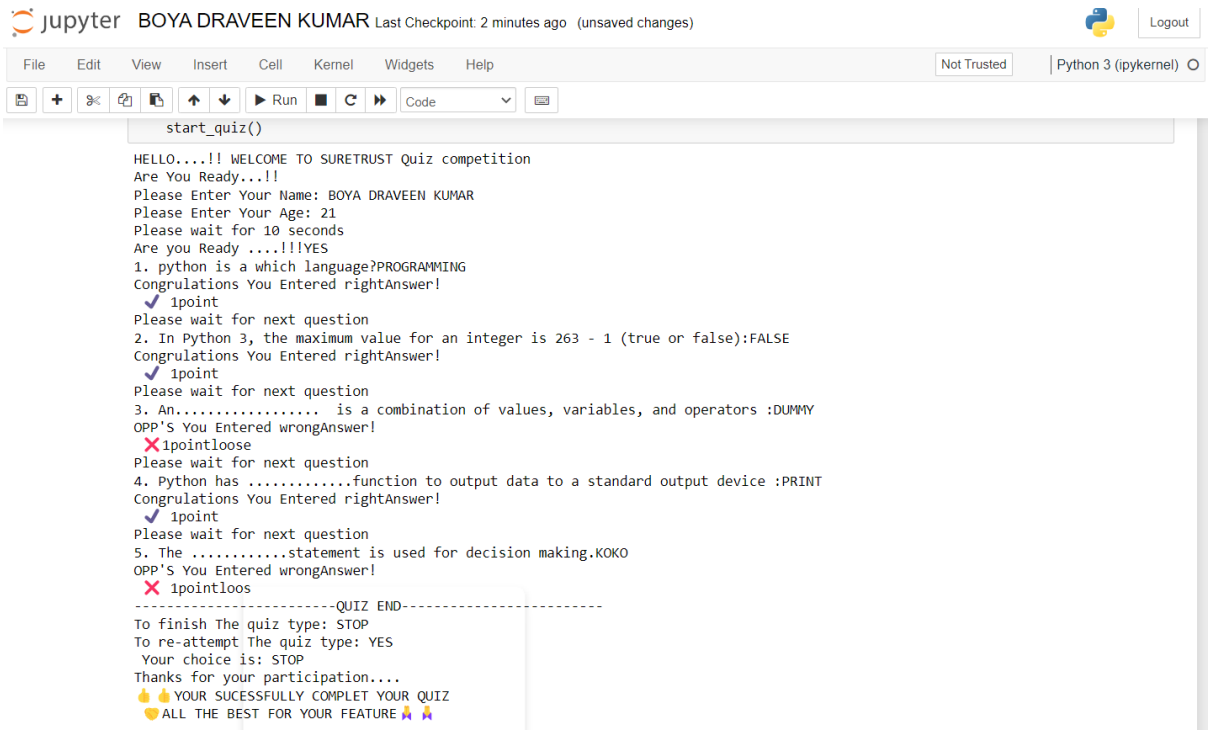
- You can have the quiz game as either MCQ based with one correct answer OR you can create a quiz game where the user inputs the answer for the question
- You can give the option of "re-attempt" for the quiz
- You are free to add any more feature



In [11]:

```
import time
import os



print("HELLO....!! WELCOME TO SURETRUST Quiz competition")
time.sleep(5)
print("Are You Ready...!!")
time.sleep(3)

NAME = input("Please Enter Your Name: ")
Age = int(input("Please Enter Your Age: "))
os.system("cls")
print("Please wait for 10 seconds")
time.sleep(10)

def start_quiz():
    ans=input("1. python is a which language?")
    if ans.lower()=='programming':
        print("Congrulations You Entered rightAnswer! \n ✓ 1point")
    else:
        print("OPP'S You Entered wrongAnswer! \n ✗ 1pointloos")
    print("Please wait for next question")
    time.sleep(5)

    ans=input("2. In Python 3, the maximum value for an integer is 263 - 1 (true or false):")
    if ans.lower()=='false':
        print("Congrulations You Entered rightAnswer! \n ✓ 1point")
    else:
        print("OPP'S You Entered wrongAnswer! \n ✗ 1pointloose")
    print("Please wait for next question")
    time.sleep(5)
    ans=input("3. An.....  is a combination of values, variables, and operators :")
    if ans.lower()=='expression':
        print("Congrulations You Entered rightAnswer! \n ✓ 1point")
    else:
```


jupyter BOYA DRAVEEN KUMAR Last Checkpoint: 8 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
print(Congrulations You Entered rightAnswer! \n ✓ 1point)
else:
print("OPP'S You Entered wrongAnswer! \n ❌1pointloose")
print("Please wait for next question")
time.sleep(5)
ans=input("4. Python has .....function to output data to a standard output device :")
if ans.lower()=='print':
print("Congrulations You Entered rightAnswer! \n ✓ 1point")
else:
print("OPP'S You Entered wrongAnswer! \n ❌ 1pointloos")
print("Please wait for next question")
time.sleep(5)
ans=input("5. The .....statement is used for decision making.")
if ans.lower()=='if':
print("Congrulations You Entered rightAnswer! \n ✓ 1point")
else:
print("OPP'S You Entered wrongAnswer! \n ❌ 1pointloos")
print("------------------------QUIZ END-------------------------")
time.sleep(5)
next = input("To finish The quiz type: STOP\nTo re-attempt The quiz type: YES\n Your choice is: ")
time.sleep(3)
if next.lower()=='yes':
print("Quiz will start within 5 seconds be patience....")
time.sleep(5)
start_quiz()
else:
print("Thanks for your participation....")
print("👍👍YOUR SUCESSFULLY COMPLET YOUR QUIZ \n 💛ALL THE BEST FOR YOUR FEATURE🙏🙏")
a=input("Are you Ready ....!!!")
if a.lower()=='yes':
start_quiz()


jupyter BOYA DRAVEEN KUMAR Last Checkpoint: 2 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
start_quiz()
HELLO....!! WELCOME TO SURETRUST Quiz competition
Are You Ready...!!
Please Enter Your Name: BOYA DRAVEEN KUMAR
Please Enter Your Age: 21
Please wait for 10 seconds
Are you Ready ....!!!YES
1. python is a which language?PROGRAMMING
Congrulations You Entered rightAnswer!
✓ 1point
Please wait for next question
2. In Python 3, the maximum value for an integer is 263 - 1 (true or false):FALSE
Congrulations You Entered rightAnswer!
✓ 1point
Please wait for next question
3. An..... is a combination of values, variables, and operators :DUMMY
OPP'S You Entered wrongAnswer!
❌1pointloose
Please wait for next question
4. Python has .....function to output data to a standard output device :PRINT
Congrulations You Entered rightAnswer!
✓ 1point
Please wait for next question
5. The .....statement is used for decision making.KOKO
OPP'S You Entered wrongAnswer!
❌ 1pointloos
------------------------QUIZ END-------------------------
To finish The quiz type: STOP
To re-attempt The quiz type: YES
Your choice is: STOP
Thanks for your participation....
👍👍YOUR SUCESSFULLY COMPLET YOUR QUIZ
💛ALL THE BEST FOR YOUR FEATURE🙏🙏

CODE:

```
import time
import os
print("HELLO....!! WELCOME TO SURETRUST  Quiz competition")
time.sleep(5)
print("Are You Ready...!!")
time.sleep(3)

NAME = input("Please Enter Your Name: ")
Age = int(input("Please Enter Your Age: "))
os.system("cls")
print("Please wait for 10 seconds")
time.sleep(10)

def start_quiz():
    ans=input("1. python is a which language?")
    if ans.lower()=='programming':
        print("Congrulations You Entered rightAnswer! \n ✔ 1point")
    else:
        print("OPP'S You Entered wrongAnswer! \n ✖ 1pointloos")
    print("Please wait for next question")
    time.sleep(5)

    ans=input("2. In Python 3, the maximum value for an integer is 263 - 1 (true or false):")
    if ans.lower()=='false':
        print("Congrulations You Entered rightAnswer! \n ✔ 1point")
```

```
else:
    print("OPP'S You Entered wrongAnswer! \n ❌ 1pointloose")
print("Please wait for next question")
time.sleep(5)
ans=input("3. An.....  is a combination of values, variables, and operators :")
if ans.lower()=='expression':
    print("Congrulations You Entered rightAnswer! \n ✔ 1point")
else:
    print("OPP'S You Entered wrongAnswer! \n ❌1pointloose")
print("Please wait for next question")
time.sleep(5)
ans=input("4. Python has .....function to output data to a standard output device :")
if ans.lower()=='print':
    print("Congrulations You Entered rightAnswer! \n ✔ 1point")
else:
    print("OPP'S You Entered wrongAnswer! \n ❌ 1pointloos")
print("Please wait for next question")
time.sleep(5)
ans=input("5. The .....statement is used for decision making.")
if ans.lower()=='if':
    print("Congrulations You Entered rightAnswer! \n ✔ 1point")
else:
    print("OPP'S You Entered wrongAnswer! \n ❌ 1pointloos")
print("------------------------QUIZ END-------------------------")
time.sleep(5)
next = input("To finish The quiz type: STOP\nTo re-attempt The quiz type: YES\n Your choice is: ")
time.sleep(3)
```

```
    if next.lower()=='yes':
        print("Quiz will start within 5 seconds be patience....")
        time.sleep(5)
        start_quiz()
    else:
        print("Thanks for your participation....")
        print("👍👍YOUR SUCESSFULLY COMPLET YOUR QUIZ \n🤝ALL THE BEST FOR YOUR FEATURE🙏🙏")
a=input("Are you Ready ....!!!")
if a.lower()=='yes':
    start_quiz()
```

INPUT/OUTPUT

```
HELLO....!! WELCOME TO SURETRUST Quiz competition
Are You Ready...!!
Please Enter Your Name: BOYA DRAVEEN KUMAR
Please Enter Your Age: 21
Please wait for 10 seconds
Are you Ready ....!!!YES
1. python is a which language?PROGRAMMING
Congrulations You Entered rightAnswer!
 ✔ 1point
Please wait for next question
2. In Python 3, the maximum value for an integer is 263 - 1 (true or false):FALSE
Congrulations You Entered rightAnswer!
 ✔ 1point
Please wait for next question
3. An.................  is a combination of values, variables, and operators :DUMM
Y
OPP'S You Entered wrongAnswer!
 ✘1pointloose
Please wait for next question
4. Python has .............function to output data to a standard output device :PRI
NT
Congrulations You Entered rightAnswer!
 ✔ 1point
Please wait for next question
5. The ............statement is used for decision making.KOKO
OPP'S You Entered wrongAnswer!
 ✘ 1pointloos
------------------------QUIZ END-------------------------
To finish The quiz type: STOP
```

```
To re-attempt The quiz type: YES
 Your choice is: STOP
Thanks for your participation....
👍👍YOUR SUCESSFULLY COMPLET YOUR QUIZ
 🤝ALL THE BEST FOR YOUR FEATURE👨‍👩‍👧‍👦
```

----------XXXXX----------

Assignment-6:

Date: 08-Aug-2022

Name of the Student: BOYA DRAVEEN KUMAR
Name of the trainer: Ms. Mayuri Gaikwad
Course: Machine Learning With Python

---

1. What is the difference between OOP and POP?

Ans:

- **OOP - Object Oriented Programing.**
- **OOP follows bottom up approach.**
- **POP -Procedural Oriented Programming.**
- **POP follows top down approach.**

2.What is Object Oriented Programming?

Ans: **Object-oriented programming is a programming paradigm based on the concept of "objects", which can contain data and code: data in the form of fields, and code, in the form of procedures. A common feature of objects is that procedures are attached to them and can access and modify the object's data fields.**

3.What is the benefit of using OOPS?

Ans**: Benefits of using oops is nothing but it provides more security, reusability, etc.**

4.What are the main features of OOPs?

Ans**: Main features of oops are: polymorphism, abstraction, encapsulation and inheritance.**

5.What is a class?

Ans**: A class is nothing but a blueprint for the object.**

6.What is an object?

Ans: **An object is nothing but an instance of class.**

7.What is inheritance?

Ans**: Inheritance is one of the properties of oops. In this technique the object of child class can access the methods of child class and also parent class.**

8.What is a superclass?

Ans: **The class from which a class inherits is called the parent or superclass. A class which inherits from a superclass is called a subclass, also called heir class or child class. Super classes are sometimes called ancestors as well.**

9.What is a subclass?

Ans: **The class which inherits other class is called base class**

10.What are the different types of inheritance?

Ans: **.Types of inheritance:**

- **Single level inheritance**
- **Multiple level inheritance**
- **Multi level inheritance**
- **Hybrid inheritance**
- **Hierarchical inheritance**

11.What are the limitations of inheritance?

Ans: **Decreases the Execution Speed: loading multiple classes because they are interdependent on each other Tightly Coupled Classes**

----------xxxxxxxxx----------

Assignment-7:

Date: 08-Aug-2022

Name of the Student: BOYA DRAVEEN KUMAR
Name of the trainer: Ms. Mayuri Gaikwad
Course: Machine Learning With Python

1.What is the difference between class and object?

Ans: **An object is an instance of a class. When a class is created, no memory is allocated. Objects are allocated memory space whenever they are created. The class has to be declared first and only once.**

2.What is super method?

Ans: **It is used inside a sub-class method definition to call a method defined in the super class. Private methods of the super-class cannot be called as**

- **It is also used by class constructors to invoke constructors of its parent class.**
- **Super keyword are not used in static Method.**

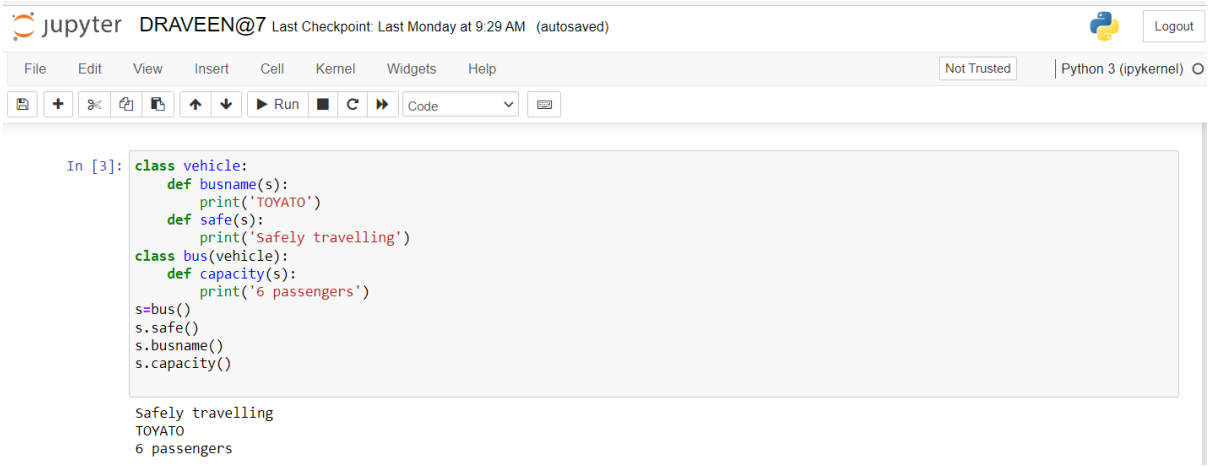
3.Can python be said to be 100% Object Oriented?

Ans**: Yes, Python is 100% Object Oriented.**

4.Create a child class Bus that will inherit all of the variables and methods of the Vehicle class

Ans:

```
class vehicle:
    def busname(s):
        print('TOYATO')
    def safe(s):
        print('Safely travelling')
class bus(vehicle):
    def capacity(s):
        print('6 passengers')
s=bus()
s.safe()
s.busname()
s.capacity()
```



5.What kind of a programming language is Python?

Ans: ***Python is an interpreted, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes. It supports multiple programming paradigms beyond object-oriented programming, such as procedural and functional programming.***

----------XXXXXX----------

Assignment-8:

Name of the Student: BOYA DRAVEEN KUMAR
Name of the trainer: Ms. Mayuri Gaikwad
Course: Machine Learning With Python

Date: 19-Aug-2022

1.What is inheritance?

Ans: Inheritance is the capability of one class to derive or inherit the properties from another class. The parent class is called base class and child class is called derived class.

2.What are the different types of inheritance?

Ans: Different types of inheritance:

- Single inheritance
- Multi-level inheritance
- Hierarchial inheritance
- Multiple inheritance

4. Explain in detail what is :

- Single inheritance
- Multiple inheritance
- Multilevel inheritance
- Hierarchical inheritance
- Hybrid inheritance

Ans:

- Single Inheritance → one parent class and one child class
- Multi-level Inheritance → child class derived from already derived baseclass
- Hierarchial Inheritance → one parent class and two child classess
- Mutiple-Inheritance → two base class and one child class

5.What are the limitations of inheritance?

Ans: Limitations of Inheritance:

- Decreases the execution speed
- Creates dependency between child and parent classess

6.What is polymorphism?

Ans: Polymorphism→performing a single task in various ways.

7.What is method overloading?

Ans: Method Overloading →Two methods having same names and different number of arguments

8. What is method overriding?

Ans: Method Overriding →Two methods having same names and same number of arguments.

9. Differentiate between overloading and overriding.

Ans:

Overloading	**Overriding**
• Defining multiples methods in same class with different parameters.	• Defining same methods in different class with same parameters.
• Method Signature is different.	• Return type + Method Signature is same.
• Checked at compile time.	• Checked at run time.
• Also called as Compile time polymorphism/Early binding/Static binding	• Run time polymorphism/Late binding/Dynamic binding.
• May or may not need inheritance.	• Must need inheritance.

Section-2

1.What is Encapsulation?

Ans: **Encapsulation** : Binding or wrapping the code and data into a single unit.

2. Differentiate between data abstraction and encapsulation.

Ans: ➲**Abstaction**:Hiding the implementation details and showing the functionalites.

➲**Encapsulation**:Binding or wrapping the code and data into a singleunit.

3. What are access specifiers in python?

Ans: **Access specifiers**→define how attributes and methods of a class can be accessed.

- ❖ Public,Private,Protected,Default

4. What is the difference between public, private and protected access modifiers?

Ans:

Public	In are defined inside class but outside methods
Private	In Private It provides high security for its own data members. It can't be inherited
Protected	In Protected it provides less security than private access modifiers

5. What is an abstract class? Can you create an instance of abstract class

Ans: An abstract class is a class that is declared abstract. It may or may not include abstract methods. It's purpose is to define how other classes should look like, i.e. what methods and properties they are expected to have.

- **Abstract classes are classes that you cannot create instances from.**
- **Use abc module to define abstract classes**.

6. What is a constructor? Describe types of constructors.

Ans: **Constructors**→they are generally used for instantiating an object. The task of constructors is to initialize to the data members of the class when an object of the class is created. In python the __int__() method is called constructor.

----------XXXXXX----------

MACHINE LEARNING:

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.
Example: Image recognition
The three machine learning types are supervised, unsupervised, and reinforcement learning.

Binary Classification Titatic:

In [1]:
```
#Dependencies
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score,confusion_matrix

%matplotlib inline
```

In [2]:
```
import warnings
warnings.filterwarnings("ignore")
```

Data Gathering

In [10]:
```
df=pd.read_csv('titanic (1).csv')
df.sample(5)
```

Out[10]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
154	155	0	3	Olsen, Mr. Ole Martin	male	NaN	0	0	Fa 265302	7.3125	NaN	S
30	31	0	1	Uruchurtu, Don. Manuel E	male	40.0	0	0	PC 17601	27.7208	NaN	C
752	753	0	3	Vande Velde, Mr. Johannes Joseph	male	33.0	0	0	345780	9.5000	NaN	S
499	500	0	3	Svensson, Mr. Olof	male	24.0	0	0	350035	7.7958	NaN	S
50	51	0	3	Panula, Master. Juha Niilo	male	7.0	4	1	3101295	39.6875	NaN	S

In [12]:
```
df.shape
```

Out[12]:
```
(891, 12)
```

In [13]:
```
df.info
```

Out[13]:
```
<bound method DataFrame.info of      PassengerId  Survived  Pclass  \
0              1         0       3
1              2         1       1
2              3         1       3
3              4         1       1
4              5         0       3
..           ...       ...     ...
886          887         0       2
887          888         1       1
888          889         0       3
889          890         1       1
890          891         0       3

                                                 Name     Sex   Age  SibSp  \
0                             Braund, Mr. Owen Harris    male  22.0      1
1    Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                              Heikkinen, Miss. Laina  female  26.0      0
3        Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                            Allen, Mr. William Henry    male  35.0      0
..                                                ...     ...   ...    ...
886                            Montvila, Rev. Juozas    male  27.0      0
887                     Graham, Miss. Margaret Edith  female  19.0      0
888         Johnston, Miss. Catherine Helen "Carrie"  female   NaN      1
889                             Behr, Mr. Karl Howell    male  26.0      0
890                               Dooley, Mr. Patrick    male  32.0      0
```

In [15]:
```
df.isnull().sum()
```

Out[15]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [16]:
```
df.duplicated().sum()
```

Out[16]:
```
0
```

In [17]:
```
df.columns
```

Out[17]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [18]:
```
columns=['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked']
```

In [ ]:
```
#Filling missing values
```

In [19]:
```
df['Age'].head() #mean
```

Out[19]:
```
0    22.0
1    38.0
2    26.0
3    35.0
4    35.0
Name: Age, dtype: float64
```

In [20]:
```
df['Age'].mean()
```

Out[20]:
```
29.69911764705882
```

In [23]:
```
df['Age'].fillna(df['Age'].mean(),inplace=True)
df['Age'].isnull().sum()
```

Out[23]:
```
0
```

In [24]:
```
df['Embarked'].head() #mode
```

Out[24]:
```
0    S
1    C
2    S
3    S
4    S
Name: Embarked, dtype: object
```

In [25]:
```
df['Embarked'].mode()
```

Out[25]:
```
0    S
Name: Embarked, dtype: object
```

In [27]:
```
df['Embarked'].fillna(df['Embarked'].mode()[0],inplace=True)
df['Embarked'].isnull().sum()
```

Out[27]:
```
0
```

In [28]:
```
df.isnull().sum()
```

Out[28]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         0
dtype: int64
```

EDA

In [33]:
```
1 columns
```

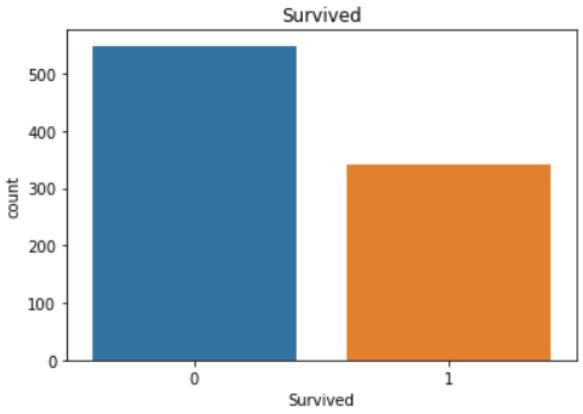
Out[33]:
```
['PassengerId',
 'Survived',
 'Pclass',
 'Name',
 'Sex',
 'Age',
 'SibSp',
 'Parch',
 'Ticket',
 'Fare',
 'Cabin',
 'Embarked']
```

In [34]:
```
1 df['Survived'].value_counts()
```

Out[34]:
```
0    549
1    342
Name: Survived, dtype: int64
```

In [35]:
```
1 sns.countplot('Survived',data=df)
2 plt.title('Survived')
```

Out[35]:
```
Text(0.5, 1.0, 'Survived')
```

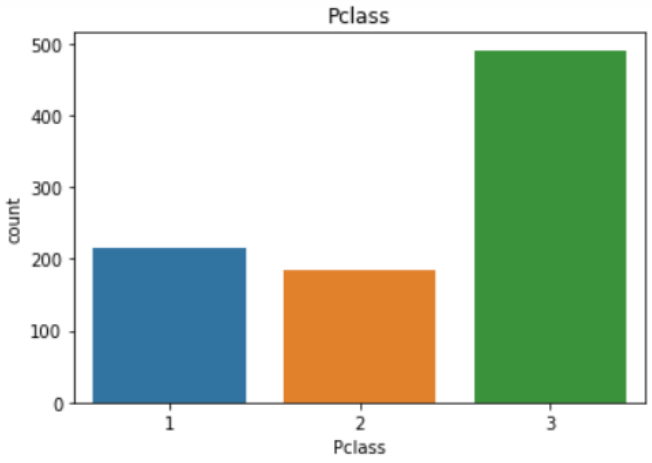


In [36]:
```
1 df['Pclass'].value_counts()
```

Out[36]:
```
3    491
1    216
2    184
Name: Pclass, dtype: int64
```

In [38]:
```
1 sns.countplot('Pclass',data=df)
2 plt.title('Pclass')
```

Out[38]:
```
Text(0.5, 1.0, 'Pclass')
```

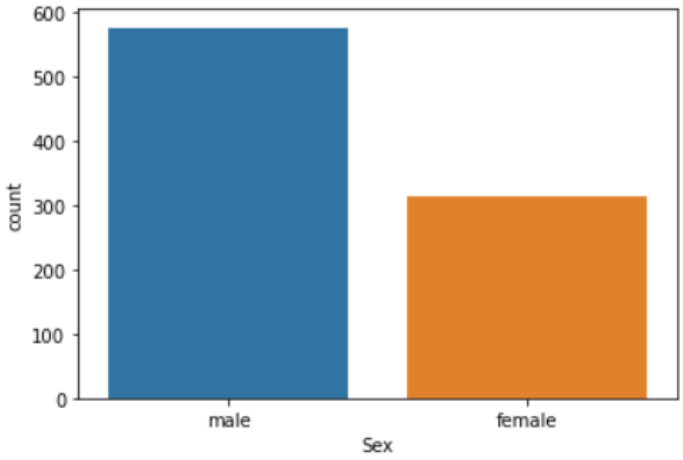


In [39]:
```
1 df['Sex'].value_counts()
```

Out[39]:
```
male      577
female    314
Name: Sex, dtype: int64
```

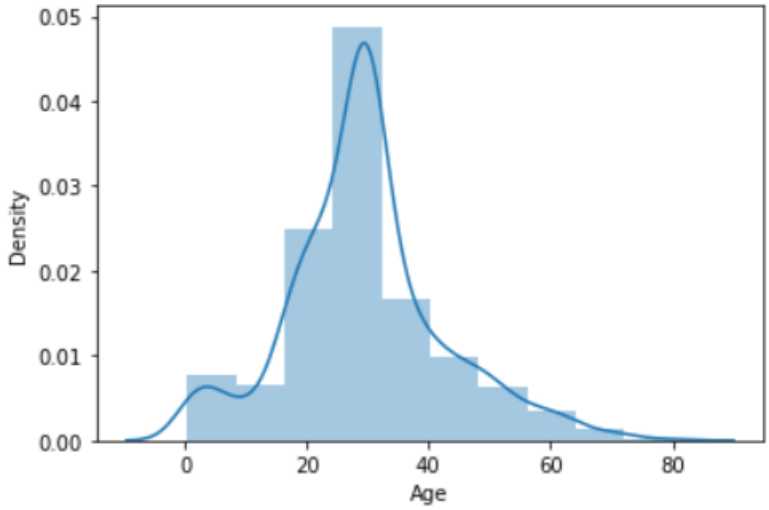
In [40]:
```
1 sns.countplot('Sex',data=df)
```

Out[40]:
```
<AxesSubplot:xlabel='Sex', ylabel='count'>
```



In [41]:
```
1 sns.distplot(df['Age'], bins=10)
```

Out[41]: <AxesSubplot:xlabel='Age', ylabel='Density'>

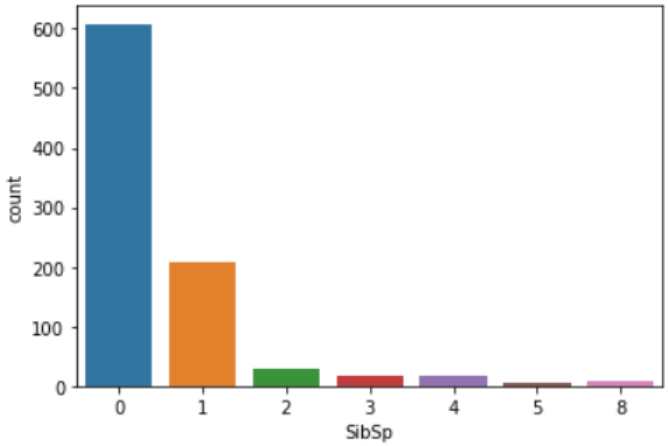


In [42]:
```
1 df['SibSp'].value_counts()
```

Out[42]:
```
0    608
1    209
2     28
4     18
3     16
8      7
5      5
Name: SibSp, dtype: int64
```

In [43]:
```
1 sns.countplot('SibSp',data=df)
```

Out[43]: <AxesSubplot:xlabel='SibSp', ylabel='count'>

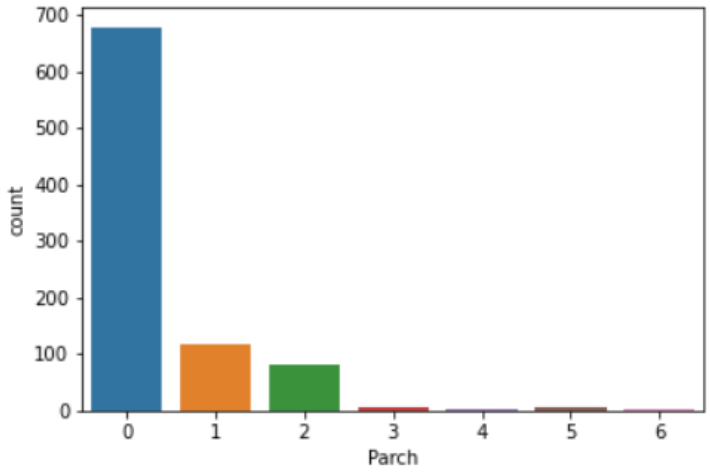


In [44]:
```
1 df['Parch'].value_counts()
```

Out[44]:
```
0    678
1    118
2     80
5      5
3      5
4      4
6      1
Name: Parch, dtype: int64
```

In [45]:
```
1 sns.countplot('Parch',data=df)
```

Out[45]: <AxesSubplot:xlabel='Parch', ylabel='count'>

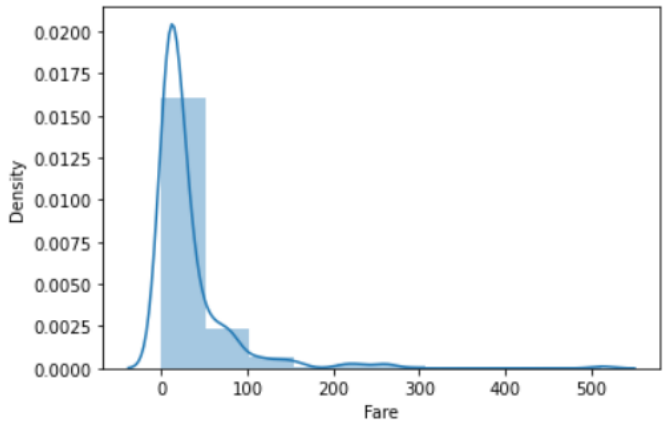


In [46]:
```
1 df['Fare'].head()
```

Out[46]:
```
0     7.2500
1    71.2833
2     7.9250
3    53.1000
4     8.0500
Name: Fare, dtype: float64
```

In [47]:
```
1 sns.distplot(df['Fare'],bins=10)
```

Out[47]: <AxesSubplot:xlabel='Fare', ylabel='Density'>

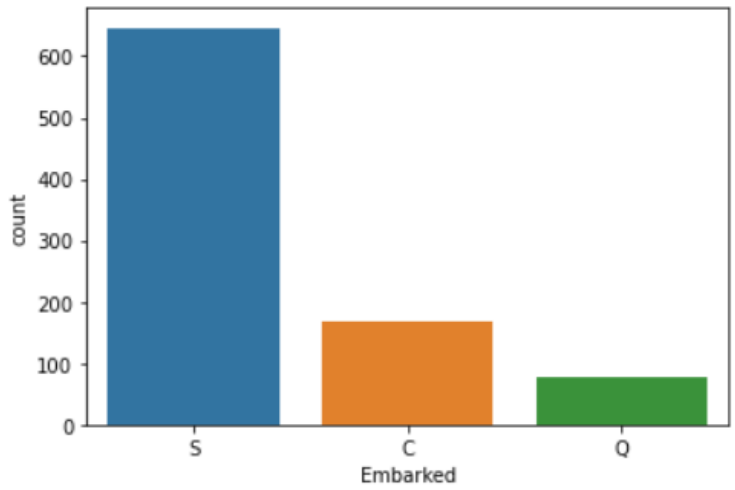


In [48]:
```
1 df["Embarked"].value_counts()
```

Out[48]:
```
S    646
C    168
Q     77
Name: Embarked, dtype: int64
```

In [49]:
```
1 sns.countplot('Embarked',data=df)
```

Out[49]: <AxesSubplot:xlabel='Embarked', ylabel='count'>



In [50]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          891 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     891 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [54]:
```
1 embark=pd.get_dummies(df['Embarked'],drop_first=True)
```

In [52]:
```
1 df['Sex'].replace({'male':0,'female':1}, inplace=True)
2 df.head(2)
```

Out[52]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	0	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	38.0	1	0	PC 17599	71.2833	C85	C

In [55]:
```
1 df.drop(columns='Embarked',axis=1,inplace=True)
```

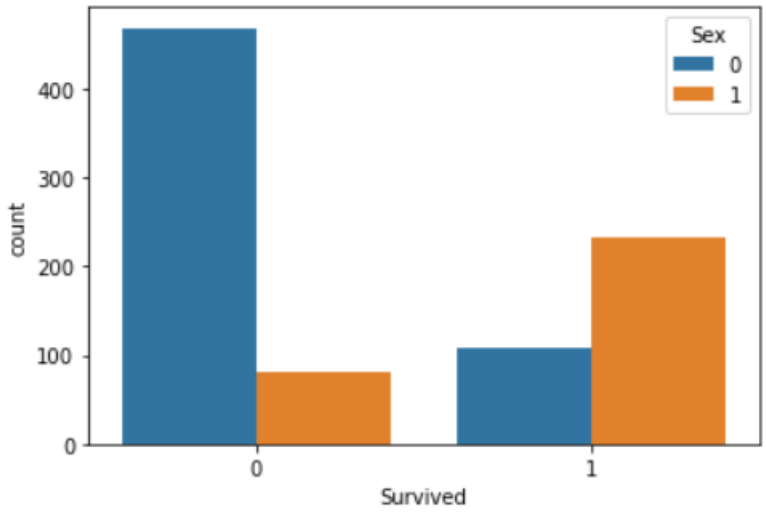
In [56]:
```
1 df.head(2)
```

Out[56]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	0	22.0	1	0	A/5 21171	7.2500	NaN
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	38.0	1	0	PC 17599	71.2833	C85

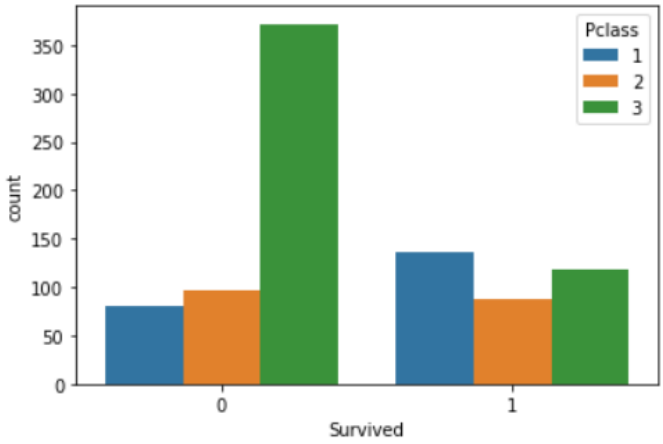
In [57]:
```
1 sns.countplot('Survived',hue='Sex',data=df)
```

Out[57]: <AxesSubplot:xlabel='Survived', ylabel='count'>



In [58]:
```
1 sns.countplot('Survived',hue='Pclass',data=df)
```

Out[58]: <AxesSubplot:xlabel='Survived', ylabel='count'>



In [59]:
```
1 family=df['SibSp']+df['Parch']
2 family.head()
```

Out[59]:
```
0    1
1    1
2    0
3    1
4    0
dtype: int64
```

In [60]:
```
1 df=pd.concat([df,family],axis=1)
```

In [61]:
```
df.drop(columns=['SibSp','Parch'],axis=1,inplace=True)
```

In [62]:
```
df.head(2)
```

Out[62]:

	PassengerId	Survived	Pclass	Name	Sex	Age	Ticket	Fare	Cabin	0
0	1	0	3	Braund, Mr. Owen Harris	0	22.0	A/5 21171	7.2500	NaN	1
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	38.0	PC 17599	71.2833	C85	1

In [ ]:
```

```

Splitting Data

In [64]:
```
X=df.drop(columns=['PassengerId','Survived','Name','Ticket','Cabin'],axis=1)
X.head(4)
```

Out[64]:

	Pclass	Sex	Age	Fare	0
0	3	0	22.0	7.2500	1
1	1	1	38.0	71.2833	1
2	3	1	26.0	7.9250	0
3	1	1	35.0	53.1000	1

In [63]:
```
y=df['Survived']
y.head()
```

Out[63]:
```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

In [ ]:
```

```

Model Training

In [65]:
```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=2)
```

In [66]:
```
lr=LogisticRegression()
```

In [67]:
```
lr.fit(X_train,y_train)
```

Out[67]:
```
LogisticRegression()
```

In [68]:
```
y_pred=lr.predict(X_test)
```

In [ ]:
```

```

Model Evaluation

In [71]:
```
a=accuracy_score(y_test,y_pred)
print(a)
```

```
0.7653631284916201
```

In [72]:
```
confusion_matrix(y_test,y_pred)
```

Out[72]:
```
array([[87, 13],
       [29, 50]], dtype=int64)
```

In [ ]:
```

```

Iris prediction:

In [2]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

data = pd.read_csv('Iris.csv')
```

In [3]:
```
data.sample(5)
```

Out[3]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
85	86	6.0	3.4	4.5	1.6	Iris-versicolor
124	125	6.7	3.3	5.7	2.1	Iris-virginica
144	145	6.7	3.3	5.7	2.5	Iris-virginica
13	14	4.3	3.0	1.1	0.1	Iris-setosa
83	84	6.0	2.7	5.1	1.6	Iris-versicolor

In [4]:
```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

In [5]:
```
1 data.describe()
```

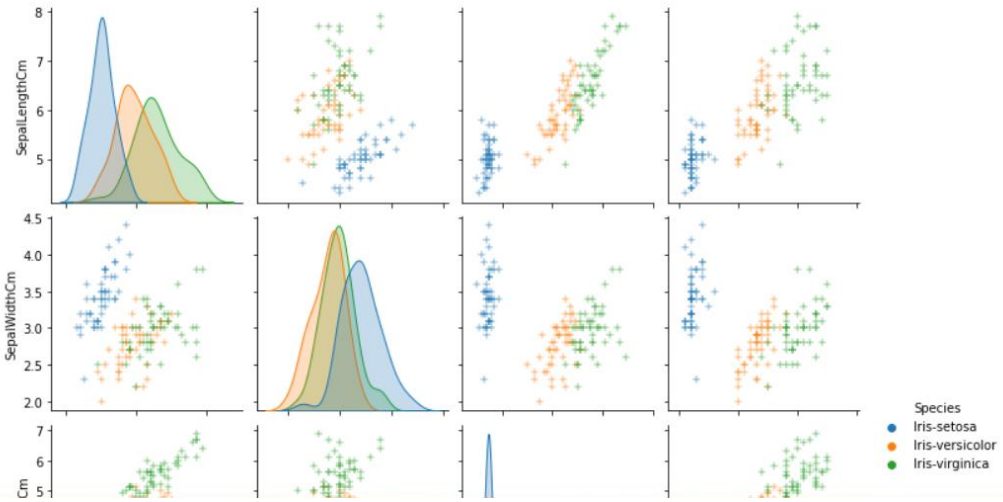
Out[5]:

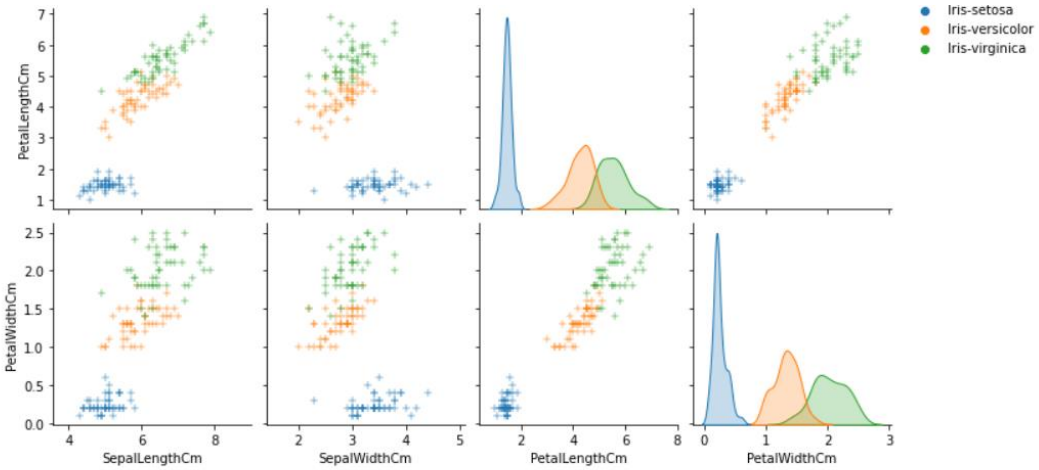
	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

In [7]:
```
1 data['Species'].value_counts()
```

Out[7]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

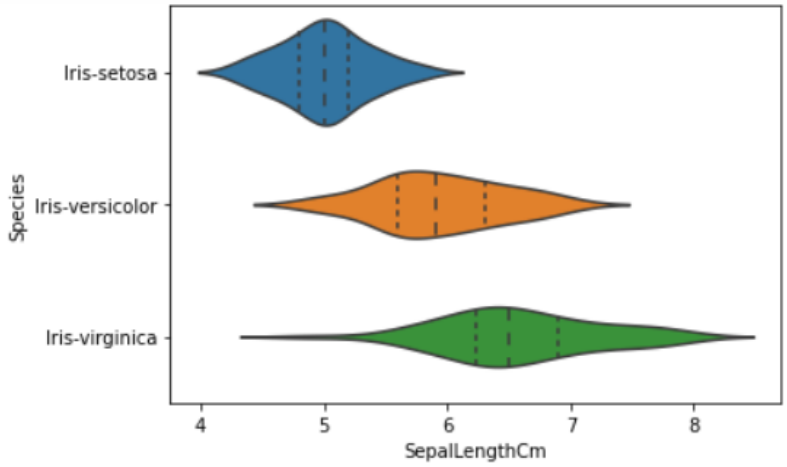
In [8]:
```
1 tmp = data.drop('Id', axis=1)
2 g = sns.pairplot(tmp, hue='Species', markers='+')
3 plt.show()
```

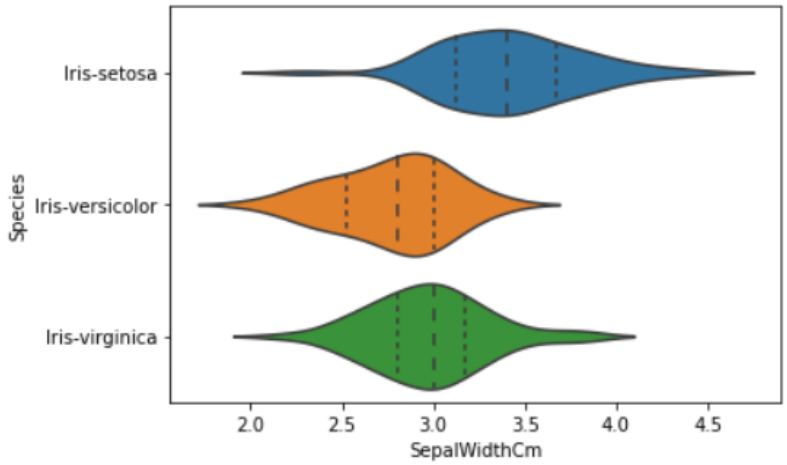


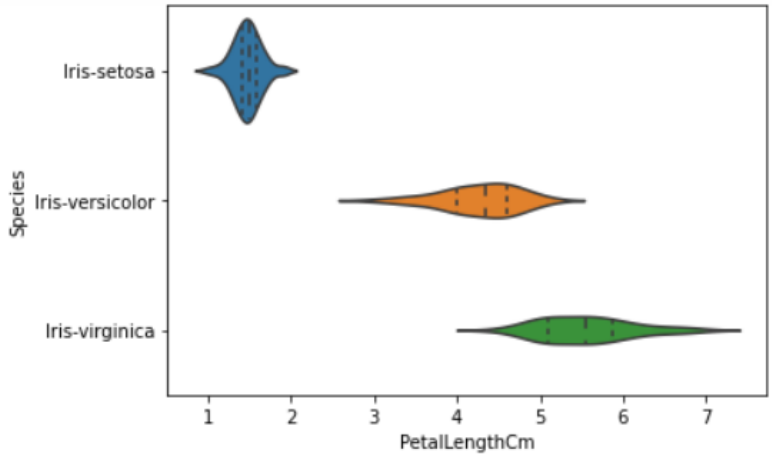


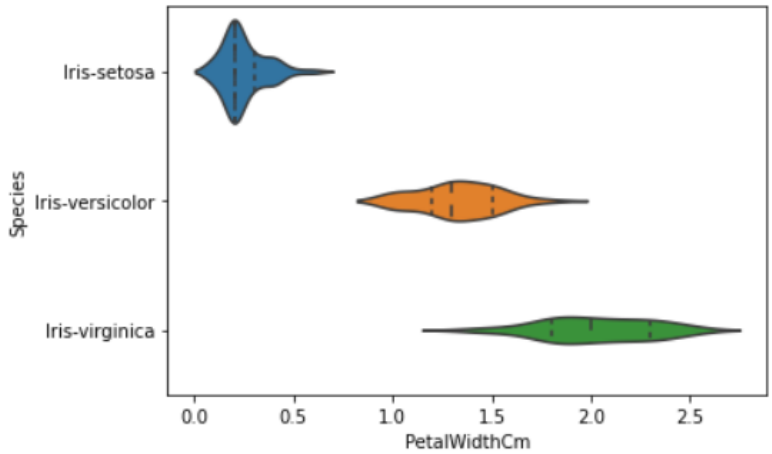
In [9]:

```
g = sns.violinplot(y='Species', x='SepalLengthCm', data=data, inner='quartile')
plt.show()
g = sns.violinplot(y='Species', x='SepalWidthCm', data=data, inner='quartile')
plt.show()
g = sns.violinplot(y='Species', x='PetalLengthCm', data=data, inner='quartile')
plt.show()
g = sns.violinplot(y='Species', x='PetalWidthCm', data=data, inner='quartile')
plt.show()
```









In [10]:
```
X = data.drop(['Id', 'Species'], axis=1)
y = data['Species']
```

In [11]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=5)
```

In [12]:
```
model1=LogisticRegression()
model2=KNeighborsClassifier()
model3=SVC()
model4=DecisionTreeClassifier()
```

In [13]:
```
model1.fit(X_train,y_train)
model2.fit(X_train,y_train)
model3.fit(X_train,y_train)
model4.fit(X_train,y_train)
```

Out[13]:
```
DecisionTreeClassifier()
```

In [14]:
```
y_pred1=model1.predict(X_test)
y_pred2=model2.predict(X_test)
y_pred3=model3.predict(X_test)
y_pred4=model4.predict(X_test)
```

In [15]:
```
score1=accuracy_score(y_test,y_pred1)
score2=accuracy_score(y_test,y_pred2)
score3=accuracy_score(y_test,y_pred3)
score4=accuracy_score(y_test,y_pred4)
```

In [16]:
```
print(score1,score2,score3,score4)
```

```
0.9736842105263158 0.9473684210526315 0.9736842105263158 0.9210526315789473
```

Multiple Linear Regression-Marks:

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error,r2_score
```

In [2]:
```
data = pd.read_csv("Student_Marks.csv")
print(data.shape)
data.head(10)
```

```
(100, 3)
```

Out[2]:

	number_courses	time_study	Marks
0	3	4.508	19.202
1	4	0.096	7.734
2	4	3.133	13.811
3	6	7.909	53.018
4	8	7.811	55.299
5	6	3.211	17.822
6	3	6.063	29.889
7	5	3.413	17.264
8	4	4.410	20.348
9	3	6.173	30.862

In [3]:
```
data.isnull().sum()
```

Out[3]:
```
number_courses    0
time_study        0
Marks             0
dtype: int64
```

In [4]:
```
data.duplicated().sum()
```

Out[4]:
```
0
```

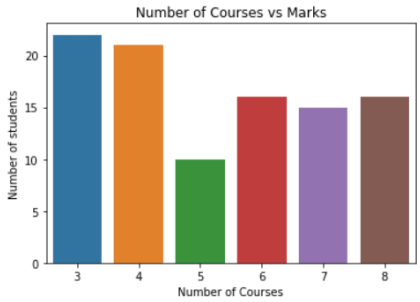
EDA

In [5]:
```
data['number_courses'].nunique()
```

Out[5]:
```
6
```

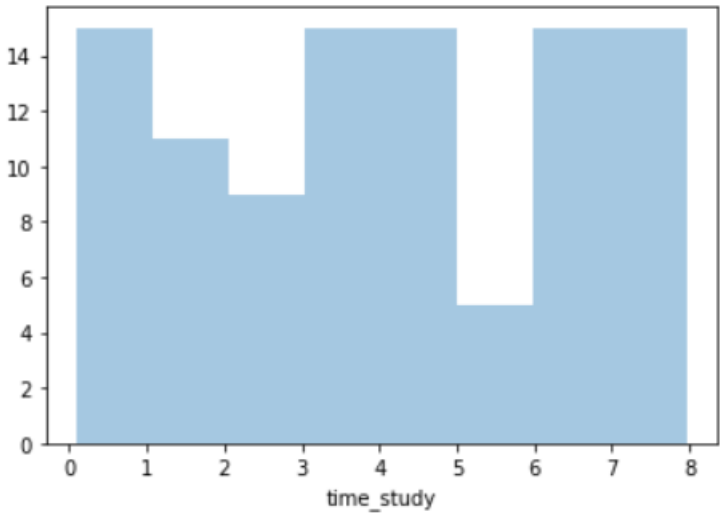
In [9]:
```
sns.countplot(data['number_courses'])
plt.xlabel("Number of Courses")
plt.ylabel("Number of students")
plt.title("Number of Courses vs Marks")
```

Out[9]:
```
Text(0.5, 1.0, 'Number of Courses vs Marks')
```



In [10]:
```
sns.distplot(data['time_study'],bins=8,kde=False)
```

Out[10]: <AxesSubplot:xlabel='time_study'>

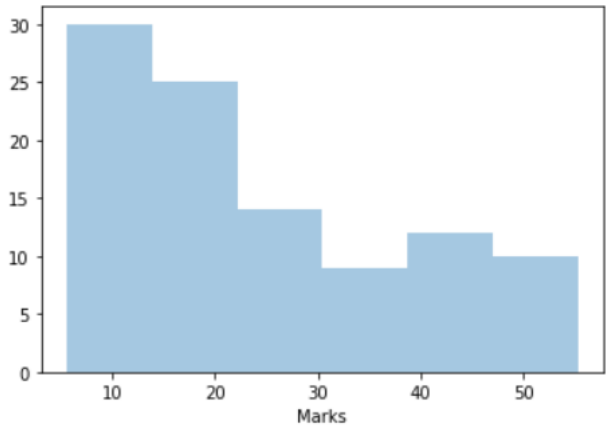


In [11]:
```
1 data['Marks'].nunique()
```

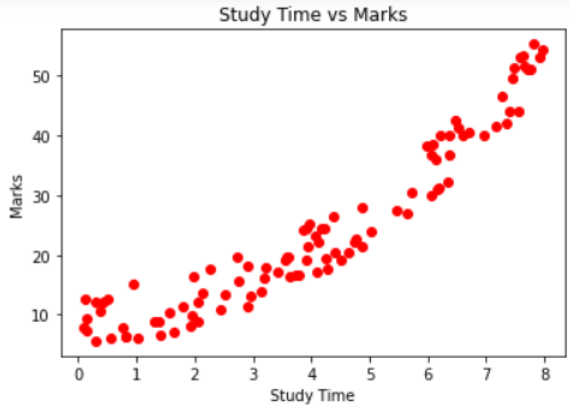
Out[11]: 100

In [12]:
```
1 sns.distplot(data['Marks'],bins=6,kde=False)
```

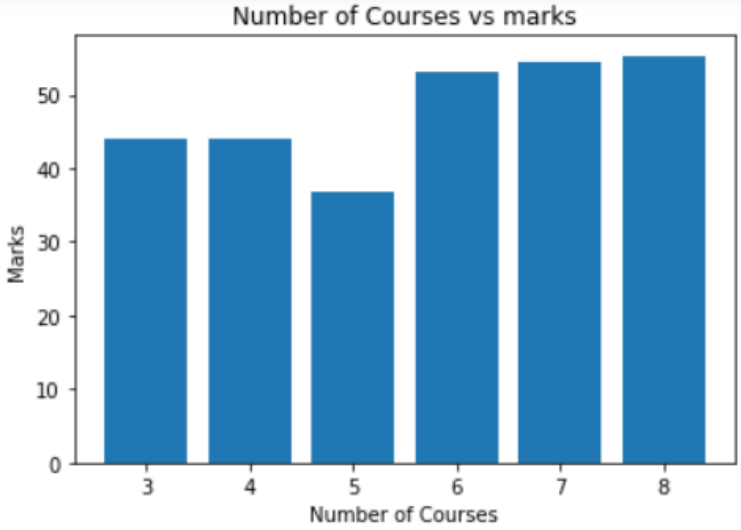
Out[12]: <AxesSubplot:xlabel='Marks'>



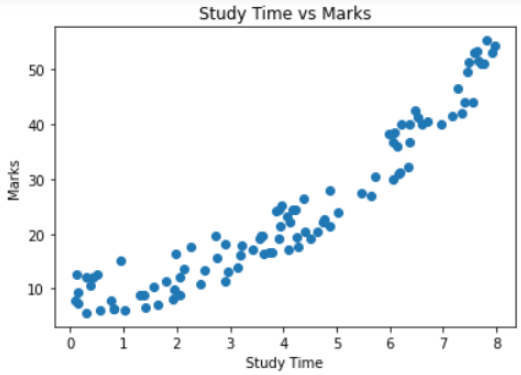
In [15]:
```
1 plt.scatter(x=data['time_study'], y=data['Marks'], c='r')
2 plt.xlabel("Study Time")
3 plt.ylabel("Marks")
4 plt.title("Study Time vs Marks");
```



In [14]:
```
plt.bar(data['number_courses'],data['Marks'])
plt.ylabel("Marks")
plt.xlabel("Number of Courses")
plt.title("Number of Courses vs marks");
```



In [16]:
```
plt.scatter(x='time_study', y='Marks',data=data)
plt.xlabel("Study Time")
plt.ylabel("Marks")
plt.title("Study Time vs Marks");
```



Model Development

In [17]:
```
# Splitting data into features and target column
X = data[['number_courses', 'time_study']]
y = data['Marks']
```

In [19]:
```
# Splitting data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Training the model

In [20]:
```
lr = LinearRegression()
```

In [21]:
```
lr.fit(X_train, y_train)
```

Out[21]:
```
LinearRegression()
```

In [22]:
```
y_pred = lr.predict(X_test)
```

Model Evaluation

In [23]:
```
mean_squared_error(y_test,y_pred)
import math
math.sqrt(mean_squared_error(y_test,y_pred))
```

Out[23]:
```
3.7683850833446595
```

In [24]:
```
r2_score(y_test,y_pred)
```

Out[24]:
```
0.9459936100591214
```

Simple Linear Regression-Salary:

Data Gathering

In [1]:
```
#Dependencies
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error,r2_score
```

In [2]:
```
df=pd.read_csv('Salary.csv')
df.shape
```

Out[2]: (35, 2)

In [3]:
```
df.head()
```

Out[3]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35 entries, 0 to 34
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   YearsExperience  35 non-null     float64
 1   Salary           35 non-null     int64
dtypes: float64(1), int64(1)
memory usage: 688.0 bytes
```

In [5]:
```
df.describe()
```

Out[5]:

	YearsExperience	Salary
count	35.000000	35.000000
mean	6.308571	83945.600000
std	3.618610	32162.673003
min	1.100000	37731.000000
25%	3.450000	57019.000000
50%	5.300000	81363.000000
75%	9.250000	113223.500000
max	13.500000	139465.000000

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
YearsExperience    0
Salary             0
dtype: int64
```

In [7]:
```
df.duplicated().sum() #df.drop_duplicate()
```

Out[7]:
```
0
```

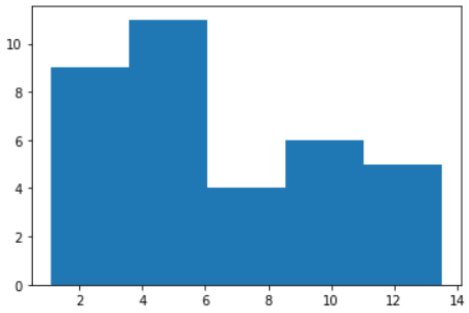
In [8]:
```
df.columns
```

Out[8]:
```
Index(['YearsExperience', 'Salary'], dtype='object')
```

EDA

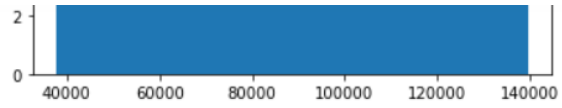
In [9]:
```
df['YearsExperience']
plt.hist(df['YearsExperience'],bins=5)
```

Out[9]:
```
(array([ 9., 11.,  4.,  6.,  5.]),
 array([ 1.1 ,  3.58,  6.06,  8.54, 11.02, 13.5 ]),
 <BarContainer object of 5 artists>)
```



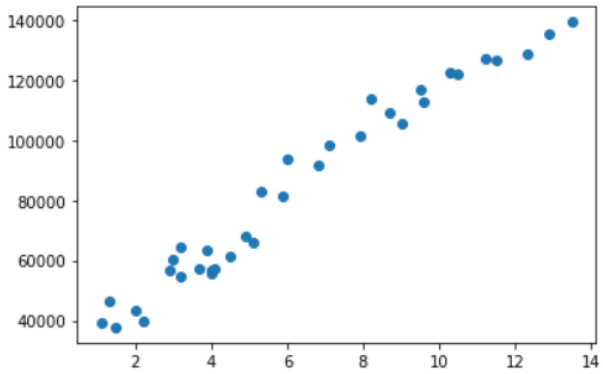
In [10]:
```
plt.hist(df['Salary'],bins=5)
```

Out[10]:
```
(array([11.,  6.,  5.,  6.,  7.]),
 array([ 37731. ,  58077.8,  78424.6,  98771.4, 119118.2, 139465. ]),
 <BarContainer object of 5 artists>)
```



In [11]:
```
1 plt.scatter(x=df['YearsExperience'],y=df['Salary'])
```

Out[11]: <matplotlib.collections.PathCollection at 0x1aabc680e20>



In [13]:
```
1 df.head()
```

Out[13]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525

Model Building

In [14]:
```
1 X=df[['YearsExperience']]
2 y=df[['Salary']]
```

In [16]:
```
1 X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random_state=5)
2 y_train
```

Out[16]:

	Salary
19	93940
23	113812
10	63218
32	128765
2	37731
6	60150
30	127345
34	139465
1	46205
11	55794
13	57081
24	109431
26	116969
25	105582
21	98273

21	98273
17	83088
15	67938
12	56957
29	121872
28	122391
33	135675
27	112635
7	54445
4	39891
8	64445
9	57189
16	66029
14	61111

In [17]:
```
1 lr=LinearRegression()
```

In [18]:
```
1 lr.fit(X_train,y_train)
```

Out[18]: LinearRegression()

In [19]:
```
1 y_pred=lr.predict(X_test)
```

In [20]:
```
1 c=lr.intercept_
2 print(c)
```

[28491.80153373]

Model Evaluation

In [22]:
```
1 mean_squared_error(y_test,y_pred)
2 import math
3 math.sqrt(mean_squared_error(y_test,y_pred))
```

Out[22]: 2661.8317840064547

In [23]:
```
1 r2_score(y_test,y_pred)
```

Out[23]: 0.9920709643930944

In [ ]:
```
1
```

PROJECTS

Graduate Admission:

Data Collection/Data Extraction

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error,r2_score
```

In [3]:
```
# Import Dataset
data = pd.read_csv("Admission_Predict.csv")
data.shape
```

Out[3]: (400, 9)

In [4]:
```
data.head(2)
```

Out[4]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76

In [5]:
```
data.columns.values
```

Out[5]:
```
array(['Serial No.', 'GRE Score', 'TOEFL Score', 'University Rating',
       'SOP', 'LOR ', 'CGPA', 'Research', 'Chance of Admit '],
      dtype=object)
```

In [6]:
```
data.drop('Serial No.', axis=1, inplace=True)
```

In [7]:
```
data.rename({'Chance of Admit ': 'Chance of Admit', 'LOR ':'LOR'}, axis=1, inplace=True)
```

Data Analysis or Data Exploration

In [8]:
```
#Let's see top 10 observation row and column wise
data.head(10)
```

Out[8]:

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	337	118	4	4.5	4.5	9.65	1	0.92
1	324	107	4	4.0	4.5	8.87	1	0.76
2	316	104	3	3.0	3.5	8.00	1	0.72
3	322	110	3	3.5	2.5	8.67	1	0.80
4	314	103	2	2.0	3.0	8.21	0	0.65
5	330	115	5	4.5	3.0	9.34	1	0.90
6	321	109	3	3.0	4.0	8.20	1	0.75
7	308	101	2	3.0	4.0	7.90	0	0.68
8	302	102	1	2.0	1.5	8.00	0	0.50
9	323	108	3	3.5	3.0	8.60	0	0.45

In [9]:
```
# Let's see the detail information of dataset
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   GRE Score          400 non-null    int64  
 1   TOEFL Score        400 non-null    int64  
 2   University Rating  400 non-null    int64  
 3   SOP                400 non-null    float64
 4   LOR                400 non-null    float64
 5   CGPA               400 non-null    float64
 6   Research           400 non-null    int64  
 7   Chance of Admit    400 non-null    float64
dtypes: float64(4), int64(4)
memory usage: 25.1 KB
```

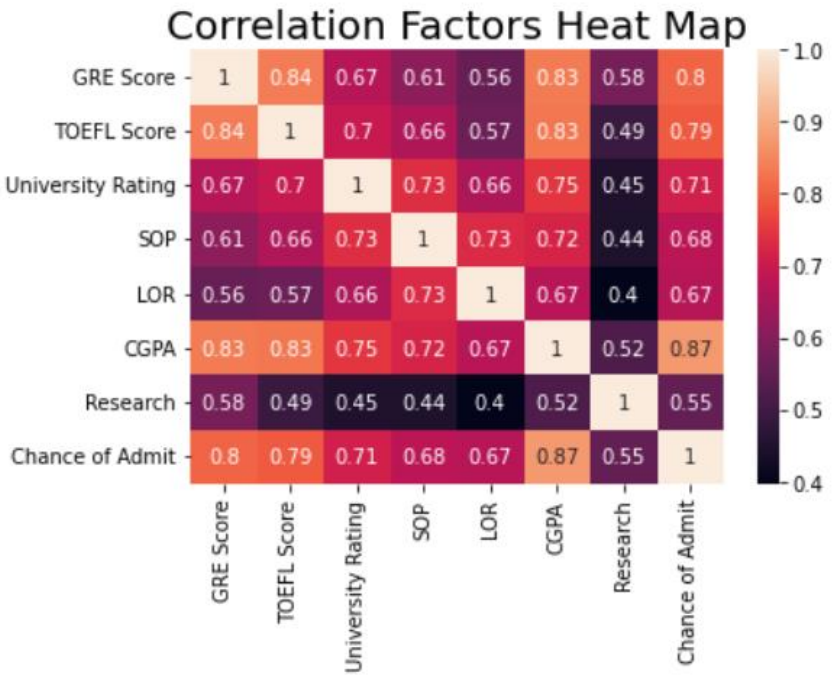
In [10]:
```
## General statistics of the data
data.describe()
```

Out[10]:

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
count	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000
mean	316.807500	107.410000	3.087500	3.400000	3.452500	8.598925	0.547500	0.724350
std	11.473646	6.069514	1.143728	1.006869	0.898478	0.596317	0.498362	0.142609
min	290.000000	92.000000	1.000000	1.000000	1.000000	6.800000	0.000000	0.340000
25%	308.000000	103.000000	2.000000	2.500000	3.000000	8.170000	0.000000	0.640000
50%	317.000000	107.000000	3.000000	3.500000	3.500000	8.610000	1.000000	0.730000
75%	325.000000	112.000000	4.000000	4.000000	4.000000	9.062500	1.000000	0.830000
max	340.000000	120.000000	5.000000	5.000000	5.000000	9.920000	1.000000	0.970000

In [11]:
```
## Correlation coeffecients heatmap
sns.heatmap(data.corr(), annot=True).set_title('Correlation Factors Heat Map', color='black', size='20')
```

Out[11]: Text(0.5, 1.0, 'Correlation Factors Heat Map')

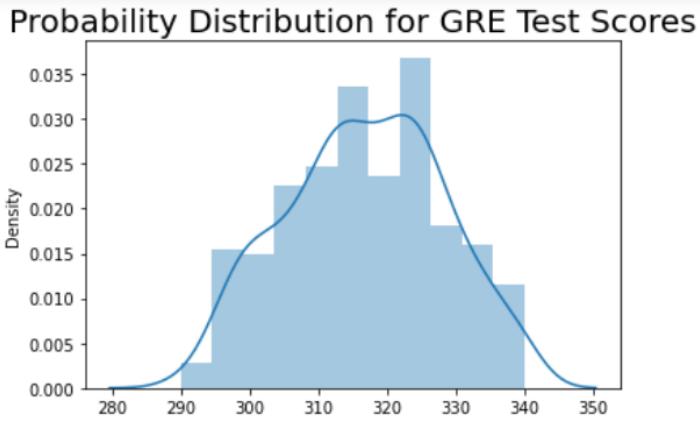


In [12]:
```
# Isolating GRE Score data
GRE = pd.DataFrame(data['GRE Score'])
GRE.describe()
```

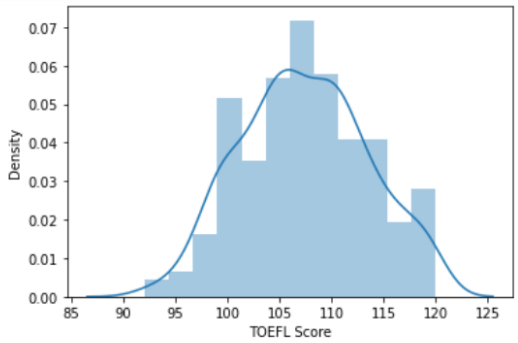
Out[12]:

	GRE Score
count	400.000000
mean	316.807500
std	11.473646
min	290.000000
25%	308.000000
50%	317.000000
75%	325.000000
max	340.000000

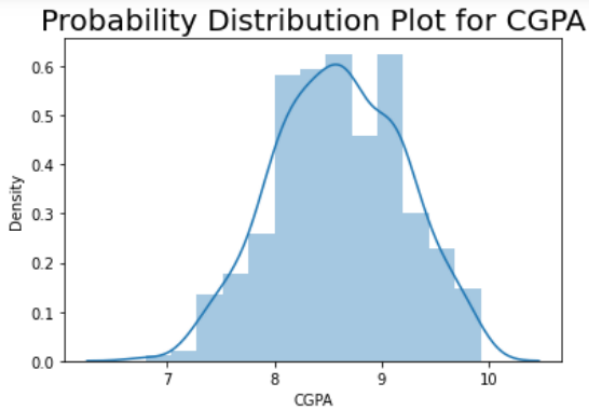
In [13]:
```
# # Probability Distribution
sns.distplot(GRE).set_title('Probability Distribution for GRE Test Scores', size='20')
plt.show()
```



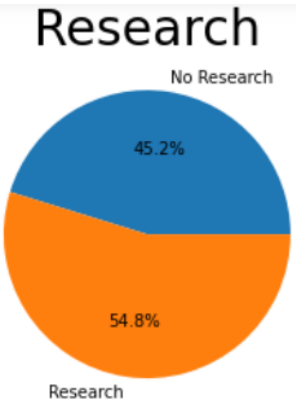
In [14]:
```
# Probability distribution for TOEFL Scores
sns.distplot(data['TOEFL Score'])
plt.show()
```



In [15]:
```
sns.distplot(data['CGPA']).set_title('Probability Distribution Plot for CGPA', size='20')
plt.show()
```



In [16]:
```
RES_Count = data.groupby(['Research']).count()
RES_Count = RES_Count['GRE Score']
RES_Count = pd.DataFrame(RES_Count)
RES_Count.rename({'GRE Score': 'Count'}, axis=1, inplace=True)
RES_Count.rename({0: 'No Research', 1:'Research'}, axis=0, inplace=True)
plt.pie(x=RES_Count['Count'], labels=RES_Count.index, autopct='%1.1f%%')
plt.title('Research', pad=5, size=30)
plt.show()
```

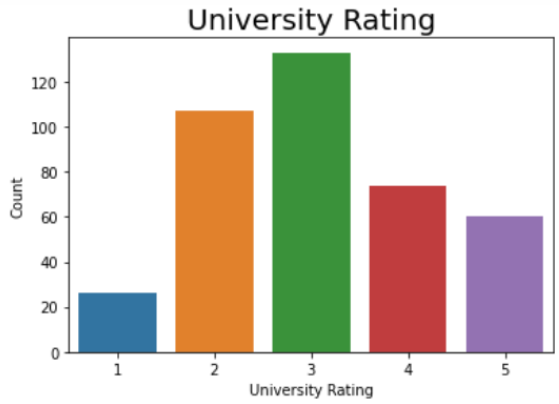


In [17]:
```
# Isolating and describing
University_Rating = data.groupby(['University Rating']).count()
University_Rating = University_Rating['GRE Score']
University_Rating = pd.DataFrame(University_Rating)
University_Rating.rename({'GRE Score': 'Count'}, inplace=True, axis=1)
University_Rating
```

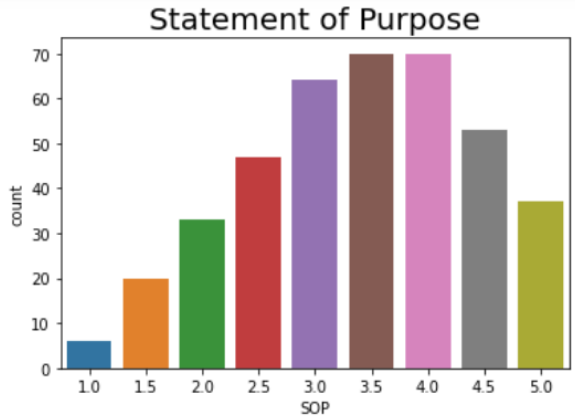
Out[17]:

University Rating	Count
1	26
2	107
3	133
4	74
5	60

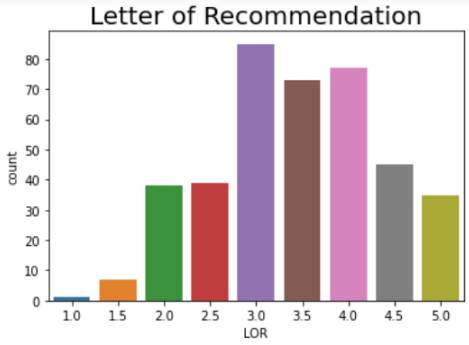
In [18]:
```
# Barplot for the distribution of the University Rating
sns.barplot(University_Rating.index, University_Rating['Count']).set_title('University Rating', size='20')
plt.show()
```



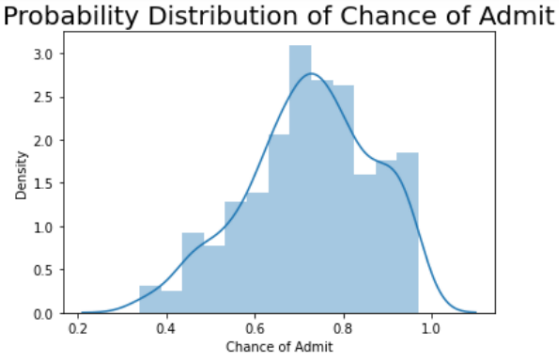
In [19]:
```
# Barplot for SOP
sns.countplot(data['SOP']).set_title('Statement of Purpose', size='20')
plt.show()
```



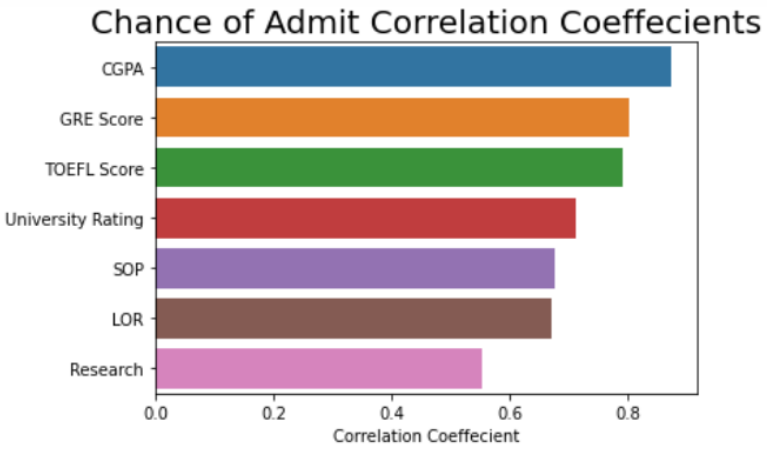
In [20]:
```
# Distribution of the LOR
sns.countplot(data['LOR']).set_title('Letter of Recommendation', size='20')
plt.show()
```



In [21]:
```
sns.distplot(data['Chance of Admit']).set_title('Probability Distribution of Chance of Admit', size='20')
plt.show()
```



In [22]:
```
COA_corr = pd.DataFrame(data.corr()['Chance of Admit'])
COA_corr.rename({'Chance of Admit': 'Correlation Coeffecient'}, axis=1, inplace=True)
COA_corr.drop('Chance of Admit', inplace=True)
COA_corr.sort_values(['Correlation Coeffecient'], ascending=False, inplace=True)
COA_corr_x = COA_corr.index
COA_corr_y = COA_corr['Correlation Coeffecient']
sns.barplot(y=COA_corr_x,x=COA_corr_y).set_title('Chance of Admit Correlation Coeffecients', size='20')
plt.show()
```



In [23]:
```
1 data.head()
```

Out[23]:

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	337	118	4	4.5	4.5	9.65	1	0.92
1	324	107	4	4.0	4.5	8.87	1	0.76
2	316	104	3	3.0	3.5	8.00	1	0.72
3	322	110	3	3.5	2.5	8.67	1	0.80
4	314	103	2	2.0	3.0	8.21	0	0.65

In [24]:
```
1 X=data.drop('Chance of Admit',axis=1)
2 y=data.iloc[:,-1]
```

In [25]:
```
1 X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=1)
```

In [26]:
```
1 sc=StandardScaler()
2 sc.fit(X_train)
3 X_train1=sc.fit_transform(X_train)
4 X_test1=sc.transform(X_test)
5 X_test1
```

Out[26]:
```
array([[-0.42668709, -0.71517247, -0.0541213 ,  0.11070689,  0.61593539,
         0.3040688 , -1.08483674],
       [-1.48581989, -1.20487687, -0.0541213 , -1.36538492, -0.49136419,
         0.10209264,  0.92179769],
       [ 0.6324457 ,  0.75394073,  0.81181954,  0.60273749,  0.0622856 ,
         0.28723745,  0.92179769],
       [ 0.6324457 , -0.06223327,  1.67776038,  0.11070689,  0.61593539,
         0.10209264,  0.92179769],
       [ 0.45592357,  0.42747113,  0.81181954,  0.60273749,  1.72323497,
         0.89316594,  0.92179769],
       [-0.77973136, -0.22546807, -0.0541213 , -0.38132372, -0.49136419,
        -0.60482393, -1.08483674],
       [-0.60320923, -0.22546807, -0.92006215,  0.11070689, -1.04501398,
        -0.45334181, -1.08483674],
       [-1.48581989, -0.87840727, -0.92006215, -1.85741553, -1.59866377,
        -1.22758377, -1.08483674],
       [-0.60320923, -1.36811167, -0.92006215, -1.85741553, -1.59866377,
        -2.18697054, -1.08483674],
       [ 2.04462277,  0.75394073,  0.81181954,  1.58679869,  1.16958518,
```

In [27]:
```
model1=LinearRegression()
model2=KNeighborsRegressor()
model3=SVR()
model4=RandomForestRegressor()
```

In [28]:
```
model1.fit(X_train1,y_train)
model2.fit(X_train1,y_train)
model3.fit(X_train1,y_train)
model4.fit(X_train1,y_train)
```

Out[28]:
```
RandomForestRegressor()
```

In [29]:
```
y_pred1 = model1.predict(X_test1)
y_pred2 = model2.predict(X_test1)
y_pred3 = model3.predict(X_test1)
y_pred4 = model4.predict(X_test1)
mse1=mean_squared_error(y_test,y_pred1)
mse2=mean_squared_error(y_test,y_pred2)
mse3=mean_squared_error(y_test,y_pred3)
mse4=mean_squared_error(y_test,y_pred4)
print('Linear Regression:',mse1)
print('KNN :',mse2)
print('SVR:',mse3)
print('Decision Tree :',mse4)
```

```
Linear Regression: 0.004442679729994739
KNN : 0.00510475
SVR: 0.005613913581346591
Decision Tree : 0.005124207624999997
```

In [30]:
```
r2_1=r2_score(y_test,y_pred1)
r2_2=r2_score(y_test,y_pred2)
r2_3=r2_score(y_test,y_pred3)
r2_4=r2_score(y_test,y_pred4)
print('Linear Regression:',r2_1)
print('KNN :',r2_2)
print('SVR:',r2_3)
print('Decision Tree :',r2_4)
```

```
Linear Regression: 0.8079043677020102
KNN : 0.779277319417683
SVR: 0.7572617553783643
Decision Tree : 0.7784359972867726
```

Wine-quality-prediction– Classification:

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

from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error,r2_score
```

In [2]:
```
wine_dataset = pd.read_csv('wine_quality.csv')
```

In [3]:
```
# number of rows ans column in the dataset
wine_dataset.shape
```

Out[3]: (1143, 13)

In [4]:
```
# first 5 rows of the dataset
wine_dataset.sample(5)
```

Out[4]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	Id
1131	6.1	0.715	0.10	2.6	0.053	13.0	27.0	0.99362	3.57	0.50	11.9	5	1582
567	7.4	0.370	0.43	2.6	0.082	18.0	82.0	0.99708	3.33	0.68	9.7	6	790
673	10.2	0.340	0.48	2.1	0.052	5.0	9.0	0.99458	3.20	0.69	12.1	7	953
430	10.1	0.650	0.37	5.1	0.110	11.0	65.0	1.00260	3.32	0.64	10.4	6	608
661	6.6	0.610	0.01	1.9	0.080	8.0	25.0	0.99746	3.69	0.73	10.5	5	934

In [5]:
```
# checking the missing value
wine_dataset.isnull().sum()
```

Out[5]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
Id                      0
dtype: int64
```

Data Analysis and Visualization

In [6]:
```
# statistical measures of the dataset
wine_dataset.describe()
```

Out[6]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	qu
count	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.00
mean	8.311111	0.531339	0.268364	2.532152	0.086933	15.615486	45.914698	0.996730	3.311015	0.657708	10.442111	5.65
std	1.747595	0.179633	0.196686	1.355917	0.047267	10.250486	32.782130	0.001925	0.156664	0.170399	1.082196	0.80
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000	3.00
25%	7.100000	0.392500	0.090000	1.900000	0.070000	7.000000	21.000000	0.995570	3.205000	0.550000	9.500000	5.00
50%	7.900000	0.520000	0.250000	2.200000	0.079000	13.000000	37.000000	0.996680	3.310000	0.620000	10.200000	6.00
75%	9.100000	0.640000	0.420000	2.600000	0.090000	21.000000	61.000000	0.997845	3.400000	0.730000	11.100000	6.00
max	15.900000	1.580000	1.000000	15.500000	0.611000	68.000000	289.000000	1.003690	4.010000	2.000000	14.900000	8.00

In [7]:
```
wine_dataset.duplicated().sum()
```

Out[7]:
```
0
```

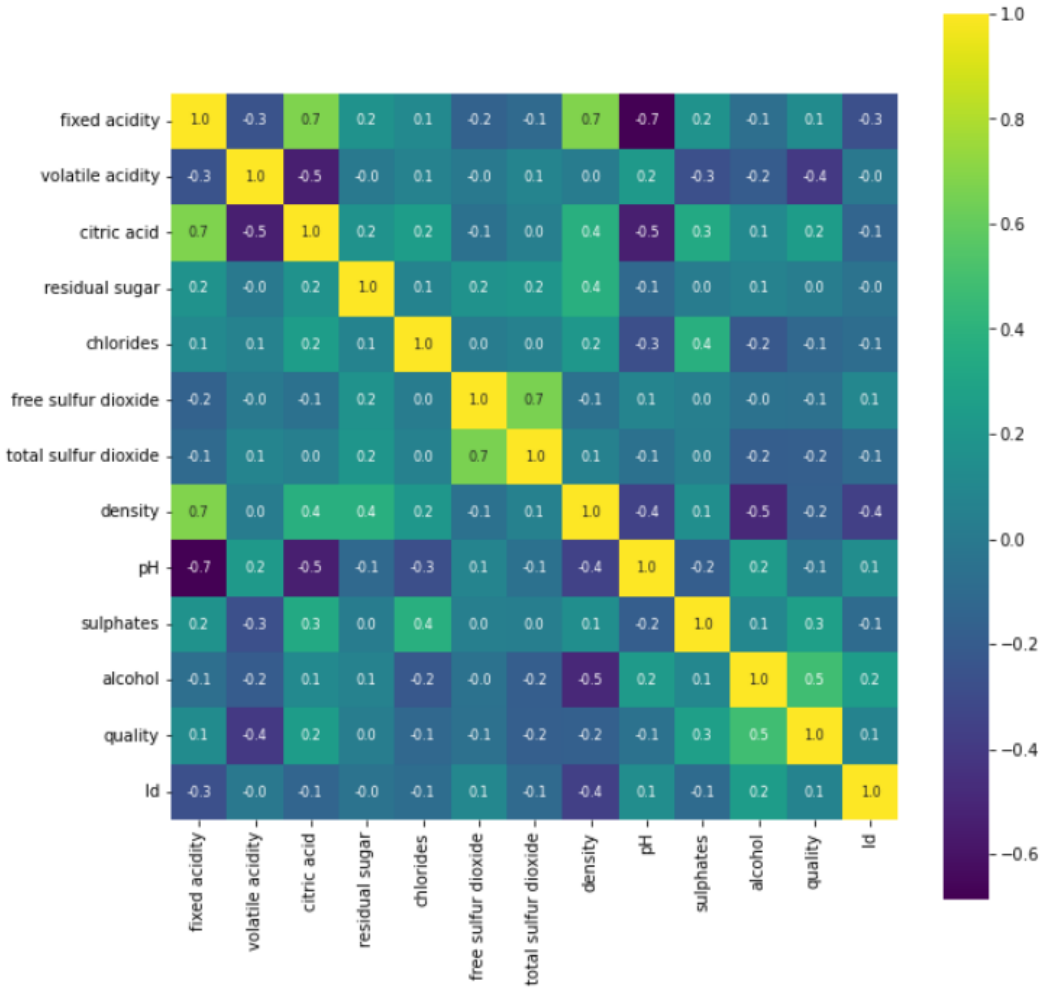
In [8]:
```
wine_dataset.isnull().sum()
```

Out[8]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
Id                      0
dtype: int64
```

In [9]:
```
correlation = wine_dataset.corr()
```

In [10]:
```
# constructing a heatmap to understand the correlation between the columns
plt.figure(figsize=(10,10))
sns.heatmap(correlation, cbar=True, square=True, fmt='.1f', annot=True, annot_kws={'size':8}, cmap='viridis')
```

Out[10]:
```
<AxesSubplot:>
```



Data Preprocessing

In [11]:
```
# seperate the data and Label
X = wine_dataset.drop('quality',axis=1)
```

Label Binarization

In [12]:
```
y = wine_dataset['quality'].apply(lambda y_value: 1 if y_value>=6.5 else 0)
y
```

Out[12]:
```
0       0
1       0
2       0
3       0
4       0
       ..
1138    0
1139    0
1140    0
1141    0
1142    0
Name: quality, Length: 1143, dtype: int64
```

Train & Test Split

In [13]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 3)
```

In [14]:
```
sc=StandardScaler()
sc.fit(X_train)
X_train1=sc.fit_transform(X_train)
X_test1=sc.transform(X_test)
X_test1
```

Out[14]:
```
array([[-0.22116534, -1.35558582,  1.10900148, ...,  0.0887674 ,
        -0.30359142,  1.54184099],
       [ 1.78468478, -0.62672611,  1.05875973, ..., -0.7439801 ,
         0.52476051,  1.14734378],
       [ 2.00755701, -0.79492451,  2.11383653, ..., -0.26812439,
         0.3406823 , -0.76939811],
       ...,
       [-1.27980845, -0.17819706, -1.35284439, ..., -1.21983582,
        -0.85582604,  1.13233573],
       [ 0.61460554, -0.73885837,  0.20464994, ..., -0.14916046,
         0.06456499, -1.48764032],
       [ 0.05742496, -1.35558582,  0.65682571, ..., -0.32760635,
         0.06456499,  1.39176053]])
```

Model Training

Randon Forest Classifier

In [15]:
```
model1 = RandomForestClassifier()
model1.fit(X_train1, y_train)
```

Out[15]:
```
RandomForestClassifier()
```

In [16]:
```
model2 = SVC()
model2.fit(X_train1, y_train)
```

Out[16]:
```
SVC()
```

In [17]:
```
model3 = LogisticRegression()
model3.fit(X_train1, y_train)
```

Out[17]:
```
LogisticRegression()
```

In [18]:
```
model4 = KNeighborsClassifier()
model4.fit(X_train1, y_train)
```

Out[18]:
```
KNeighborsClassifier()
```

In [19]:
```
# accuracy on test data
y_pred1 = model1.predict(X_test1)
y_pred2 = model2.predict(X_test1)
```

```
y_pred3 = model3.predict(X_test1)
y_pred4 = model4.predict(X_test1)
score1=accuracy_score(y_test,y_pred1)
score2=accuracy_score(y_test,y_pred2)
score3=accuracy_score(y_test,y_pred3)
score4=accuracy_score(y_test,y_pred4)
print('Random Forest :',score1*100)
print('SVM :',score2*100)
print('Linear Regression :',score3*100)
print('KNN :',score4*100)
```

```
Random Forest : 90.39301310043668
SVM : 88.64628820960698
Linear Regression : 88.20960698689956
KNN : 85.58951965065502
```

wine-quality-prediction Regression:

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

from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error,r2_score
```

In [2]:
```
wine_dataset = pd.read_csv('wine_quality.csv')
```

In [3]:
```
# number of rows ans column in the dataset
wine_dataset.shape
```

Out[3]: (1143, 13)

In [4]:
```
# first 5 rows of the dataset
wine_dataset.sample(5)
```

Out[4]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	Id
221	10.3	0.410	0.42	2.4	0.213	6.0	14.0	0.99940	3.19	0.62	9.5	6	307
733	6.9	0.440	0.00	1.4	0.070	32.0	38.0	0.99438	3.32	0.58	11.4	6	1045
507	8.9	0.565	0.34	3.0	0.093	16.0	112.0	0.99980	3.38	0.61	9.5	5	711
66	5.0	1.020	0.04	1.4	0.045	41.0	85.0	0.99380	3.75	0.48	10.5	4	94
982	7.1	0.755	0.15	1.8	0.107	20.0	84.0	0.99593	3.19	0.50	9.5	5	1384

In [5]:
```
# checking the missing value
wine_dataset.isnull().sum()
```

Out[5]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
Id                      0
dtype: int64
```

Data Analysis and Visualization

In [6]:
```
# statistical measures of the dataset
wine_dataset.describe()
```

Out[6]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	qu
count	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.00
mean	8.311111	0.531339	0.268364	2.532152	0.086933	15.615486	45.914698	0.996730	3.311015	0.657708	10.442111	5.65
std	1.747595	0.179633	0.196686	1.355917	0.047267	10.250486	32.782130	0.001925	0.156664	0.170399	1.082196	0.80
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000	3.00
25%	7.100000	0.392500	0.090000	1.900000	0.070000	7.000000	21.000000	0.995570	3.205000	0.550000	9.500000	5.00
50%	7.900000	0.520000	0.250000	2.200000	0.079000	13.000000	37.000000	0.996680	3.310000	0.620000	10.200000	6.00
75%	9.100000	0.640000	0.420000	2.600000	0.090000	21.000000	61.000000	0.997845	3.400000	0.730000	11.100000	6.00
max	15.900000	1.580000	1.000000	15.500000	0.611000	68.000000	289.000000	1.003690	4.010000	2.000000	14.900000	8.00

In [7]:
```
wine_dataset.duplicated().sum()
```

Out[7]:
```
0
```

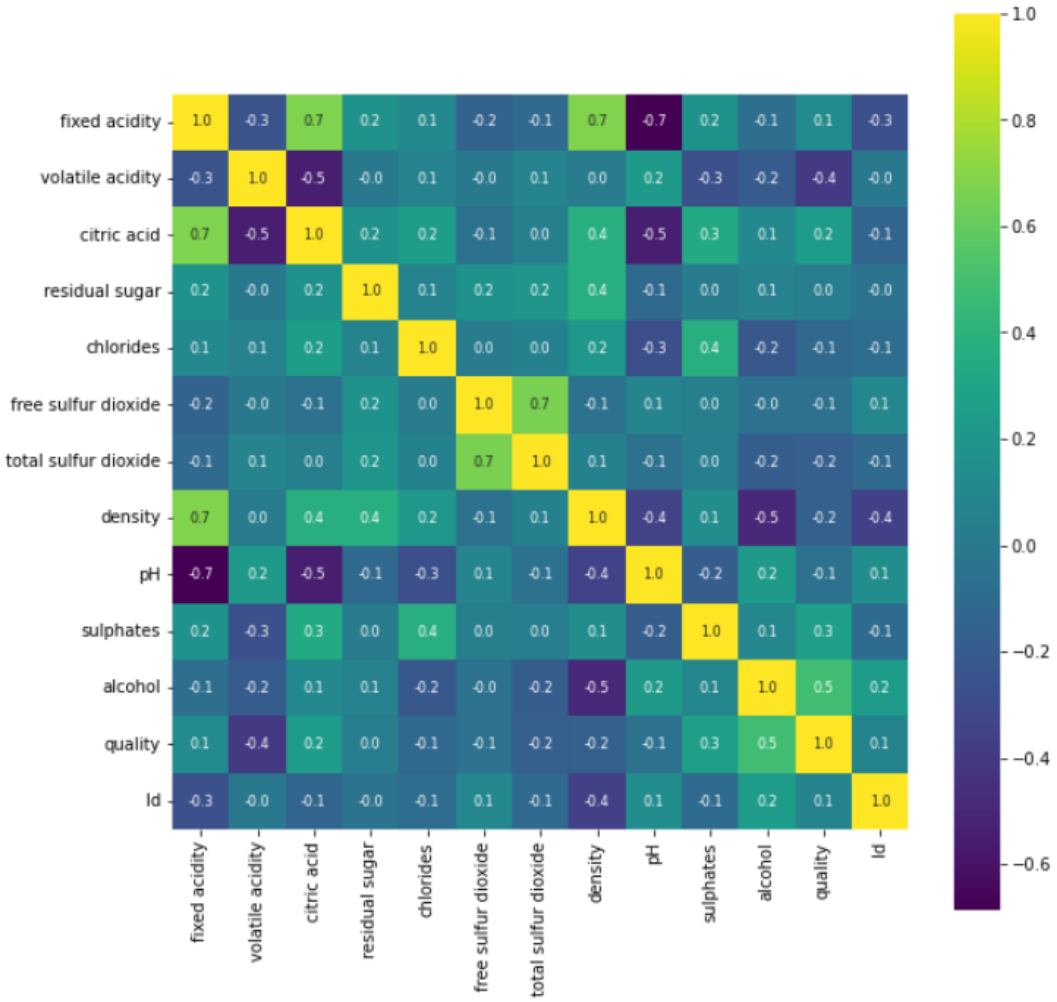
In [8]:
```
wine_dataset.isnull().sum()
```

Out[8]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
Id                      0
dtype: int64
```

In [9]:
```
correlation = wine_dataset.corr()
```

In [10]:
```
# constructing a heatmap to understand the correlation between the columns
plt.figure(figsize=(10,10))
sns.heatmap(correlation, cbar=True, square=True, fmt='.1f', annot=True, annot_kws={'size':8}, cmap='viridis')
```

Out[10]:
```
<AxesSubplot:>
```



Data Preprocessing

In [11]:
```
# seperate the data and Label
X = wine_dataset.drop('quality',axis=1)
```

Label Binarization

In [12]:
```
#y = wine_dataset['quality'].apply(lambda y_value: 1 if y_value>=6.5 else 0)
y=wine_dataset['quality']
```

Train & Test Split

In [13]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 3)
```

In [14]:
```
sc=StandardScaler()
sc.fit(X_train)
X_train1=sc.fit_transform(X_train)
X_test1=sc.transform(X_test)
X_test1
```

Out[14]:
```
array([[-0.22116534, -1.35558582,  1.10900148, ...,  0.0887674 ,
        -0.30359142,  1.54184099],
       [ 1.78468478, -0.62672611,  1.05875973, ..., -0.7439801 ,
         0.52476051,  1.14734378],
       [ 2.00755701, -0.79492451,  2.11383653, ..., -0.26812439,
         0.3406823 , -0.76939811],
       ...,
       [-1.27980845, -0.17819706, -1.35284439, ..., -1.21983582,
        -0.85582604,  1.13233573],
       [ 0.61460554, -0.73885837,  0.20464994, ..., -0.14916046,
         0.06456499, -1.48764032],
       [ 0.05742496, -1.35558582,  0.65682571, ..., -0.32760635,
         0.06456499,  1.39176053]])
```

Model Training

Randon Forest Classifier

In [15]:
```
1 model1 = RandomForestRegressor()
2 model1.fit(X_train1, y_train)
```

Out[15]:
```
RandomForestRegressor()
```

In [16]:
```
1 model2 = SVR()
2 model2.fit(X_train1, y_train)
```

Out[16]:
```
SVR()
```

In [17]:
```
model3 = LinearRegression()
model3.fit(X_train1, y_train)
```

Out[17]:
```
LinearRegression()
```

In [18]:
```
model4 = KNeighborsRegressor()
model4.fit(X_train1, y_train)
```

Out[18]:
```
KNeighborsRegressor()
```

In [19]:
```
# accuracy on test data
y_pred1 = model1.predict(X_test1)
y_pred2 = model2.predict(X_test1)
y_pred3 = model3.predict(X_test1)
y_pred4 = model4.predict(X_test1)
mse1=mean_squared_error(y_test,y_pred1)
mse2=mean_squared_error(y_test,y_pred2)
mse3=mean_squared_error(y_test,y_pred3)
mse4=mean_squared_error(y_test,y_pred4)
print('Random Forest :',mse1)
print('SVM :',mse2)
print('Linear Regression :',mse3)
print('KNN :',mse4)
```

```
Random Forest : 0.3992598253275109
SVM : 0.40744545599567106
Linear Regression : 0.44439945765165234
KNN : 0.5189519650655022
```

----------XXXXXX----------

3. Students Feedback

I am very thankful for this opportunity given by the SURE Trust. This course is very thorough and detailed. As an IT specialist, I have had no exposure to formal research and needed to understand the process. Now I can clearly and confidently say that I can perform good research and obtain formal information and data on any topic, as opposed to just surfing the internet for genuine knowledge.

4. Uniqueness of the course

- Concepts are learnt theoretically and practically.
- Faculty is committed in delivering the Couse
- Thought provoking assignments will be given
- Preparation of course report, which will help us to refer inthe future
- Concentrate on each and every student in the course

5. Concluding Remarks

A lot of experience, knowledge and exposure that I have handy. All disclosures were awakened myself in a boost of self-confidence to face life more challenging now. Practical is a complement to the science or theory learned. I conclude that the SURE TRUST training program has provided many benefits to students. I really proud to be a student