## 1. Check if a number belongs to the Fibonacci sequence

```
num=int(input("Enter the number you want to check\n"))
c=0
a=0
b=1
print("fibonnaci series:")
print(a)
if num==0 or num==1:
        print("Yes")
else:
        while c<num:
                c=a+b
                b=a
                a=c
                print(c)
 if c==num:
        print("Yes. {} is a fibonnaci number".format(num))
 else:
        print("no. {} is NOT a fibonnaci number".format(num))
```

**OUTPUT:**

```
Enter the number you want to check:
7
fibonnaci series:
0
1
1
2
3
5
8
no. 7 is NOT a fibonnaci number
```

2.Solve the Quadratic equation

```
import cmath
a=int(input("enter a"))
b=int(input("enter b"))
c=int(input("enter c"))
#Calculate discriminant
d=(b**2)-(4*a*c)
#find two solutions
sol1=(-b-cmath.sqrt(d))/(2*a)
sol2=(-b+cmath.sqrt(d))/(2*a)
print("the solution are {0} and {1}".format(sol1,sol2))
if d>0:
print("real and different root")
elif d==0:
print("real and same roots")
else:
print("complex roots")
```

**Output:**

- Enter a 1
  Enter b 2
  Enter c 3
  The solution are (-1-1.4142135623730951j) and (-1+1.4142135623730951j)
  complex roots
- enter a 1
  enter b -5
  enter c 6
  the solution are (2+0j) and (3+0j)
  real and different root
- enter a 1
  enter b -4
  enter c 4
  the solution are (2+0j) and (2+0j)
  real and same roots

3.Find the sum of n natural numbers

```
n=int(input("enter a number:"))
sum=0
while(n>0):
    sum=sum+n
    n=n-1
print("The sum of n natural number=",sum)
```

**Output:**

- enter a number:10

  The sum of n natural number= 55

- enter a number:7

  The sum of n natural number= 28

**4. Display a multiplication table**

```
num=int(input('Display multiplication table of:'))
for i in range(1,11):
   print(num, 'X', i, '=', num*i)
```

**OUTPUT:**

```
Display multiplication table of:12
12 X 1 = 12
12 X 2 = 24
12 X 3 = 36
12 X 4 = 48
12 X 5 = 60
12 X 6 = 72
12 X 7 = 84
12 X 8 = 96
12 X 9 = 108
12 X 10 = 120
```

5. Check if a given number is a Prime or Not.

```
n=int(input("enter n"))
for i in range(2,n):
    if(n%i==0):
print("not a prime")
        break
else:
print("prime number")
```

Output:

- enter n 3
  prime number
- enter n 10
  not a prime

6. Implement a sequential search

```
Numlist=[3,1,4,7,4,8,5,0,7,9,6,87,65,44,33,55,66,877,88]
print("list items=",Numlist)
searchnum=int(input("enter a number to search"))
found=False
for i in range(len(Numlist)):
    if Numlist[i]==searchnum:
        found=True
        print(searchnum,"is present in thye list at the index",i)
        break
if found==False:
    print(searchnum,"is not present in the list")
```

**Output:**

```
list items= [3, 1, 4, 7, 4, 8, 5, 0, 7, 9, 6, 87, 65, 44, 33, 55, 66, 877, 88]
enter a number to search 4
4 is present in thye list at the index 2
```

```
list items= [3, 1, 4, 7, 4, 8, 5, 0, 7, 9, 6, 87, 65, 44, 33, 55, 66, 877, 88]
enter a number to search 999
999 is not present in the list
```

**7. Create a calculator program**

```
# Function to add two numbers
def add(num1, num2):
        return num1 + num2

# Function to subtract two numbers
def subtract(num1, num2):
        return num1 - num2

# Function to multiply two numbers
def multiply(num1, num2):
        return num1 * num2

# Function to divide two numbers
def divide(num1, num2):
   if num2 == 0:
      print('Number cannot divided by zero')
      exit()
   else:
      return num1 / num2
print("Please select operation -\n" \
                "1. Add\n" \
                "2. Subtract\n" \
                "3. Multiply\n" \
                "4. Divide\n")
# Take input from the user
select = int(input("Select operations form 1, 2, 3, 4 :"))
if select > 4:
   print('invalid choice')
   exit()

number_1 = int(input("Enter first number: "))
number_2 = int(input("Enter second number: "))
if select == 1:
        print(number_1, "+", number_2, "=",add(number_1, number_2))
elif select == 2:
        print(number_1, "-", number_2, "=",subtract(number_1, number_2))
elif select == 3:
        print(number_1, "*", number_2, "=",multiply(number_1, number_2))
elif select == 4:
        print(number_1, "/", number_2, "=",divide(number_1, number_2))
else:
        print("Invalid input")
```







OUTPUT:

```
Please select operation -
1. Add
2. Subtract
3. Multiply
4. Divide

Select operations form 1, 2, 3, 4 :1
Enter first number: 5
Enter second number: 9
5 + 9 = 14
```

## 8. Explore String functions.

```
s = "hello world1"
length_of_s = len(s)
print(f"Length of the string: {length_of_s}")
uppercase_s = s.upper()
print(f"String in uppercase: {uppercase_s}")
lowercase_s = s.lower()
print(f"String in lowercase: {lowercase_s}")
count_of_l = s.count("l")
print(f"Number of occurrences of 'l' in the string: {count_of_l}")
index_of_o = s.index("o")
print(f"Index of the first occurrence of 'o': {index_of_o}")
stripped_s = s.strip()
print(f"String with leading and trailing whitespace removed: {stripped_s}")
```



## Output:

```
Length of the string: 12
String in uppercase: HELLO WORLD1
String in lowercase: hello world1
Number of occurrences of 'l' in the string: 3
Index of the first occurrence of 'o': 4
String with leading and trailing whitespace removed: hello world1

Process finished with exit code 0
```

## 9. Implement selection sort

```
a=[]
n=int(input("Number of elements in array:"))
print('Enter', n ,'elements')
for i in range(0,n):
  l=int(input())
  a.append(l)
print('Elements before sorting')
print(a)

for i in range(n):
      min = i
      for j in range(i+1, n):
         if a[j]<a[min]:
            min = j

      a[i], a[min] = a[min], a[i]
print('Elements after sorting')
print(a)
```

**OUTPUT:**

```
Number of elements in array:5
Enter 5 elements
56
96
48
23
21
Elements before sorting
[56, 96, 48, 23, 21]
Elements after sorting
[21, 23, 48, 56, 96]
```

## 10. Implement Stack

```
stack=[]
s=int(input('Enter the Stack size :'))

while True:
  print('1.push\n2.pop\n3.Display\n4.Exit\n')
  ch=int(input('Enter the choice:'))
  if ch == 1:
    if(len(stack) == s):
      print("Stack Overflow")
    else:
      item=int(input('Enter the item to push:'))
      stack.append(item)
      print("The items contains in the stack")
      print(stack)
  elif ch == 2:
    if(len(stack) == 0):
      print("Stack Underflow")
    else:
      print(stack.pop())
      print("The items contains in the stack")
      print(stack)
  elif ch == 3:
    if(len(stack) == 0):
      print("Stack is Empty to Display")
    else:
      print(stack)
  elif ch == 4:
    exit()
  else:
    print('Invalid Input')
```







**OUTPUT:**

```
Enter the Stack size :5
1.push
2.pop
3.Display
4.Exit

Enter the choice:1
Enter the item to push:23
The items contains in the stack
[23]
1.push
2.pop
3.Display
4.Exit

Enter the choice:1
Enter the item to push:41
The items contains in the stack
[23, 41]
1.push
2.pop
3.Display
4.Exit

Enter the choice:3
[23, 41]
```

**11.Read and write into a file**

```
file1 = open("myfile.txt","w")
L = ["BCA Department\n","Shree Medha\n","Ballari\n"]

# \n is placed to indicate EOL (End of Line)
file1.write("Hello \n")
file1.writelines(L)
file1.close() #to change file access modes

file1 = open("myfile.txt","r")

print("Output of Read function is ")
print(file1.read())
print()
```

**OUTPUT:**

```
Output of Read function is
Hello
BCA Department
Shree Medha
Ballari
```

## 1. Demonstrate usage of basic regular expression

```
import re
# Define a pattern to match email addresses
email_pattern = r'\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b'
# Test if a string matches the pattern
email = 'example@example.com'
if re.match(email_pattern, email):
   print('{} matches the email pattern'.format(email))
else:
   print('{} does not match the email pattern'.format(email))
# Find all matches of the pattern in a string
text = 'Contact us by email at info@example.com or call us at 123-456-7890'
matches = re.findall(email_pattern, text)
print('Email addresses found in text: {}'.format(matches))
```

**OUTPUT:**

```
example@example.com matches the email pattern
Email addresses found in text: ['info@example.com']
```

## 2. Demonstrate use of advanced regular expressions for data validation.

```
import re
n = input("Enter mobile number: ")
r = re.fullmatch('[6-9][0-9]{9}', n)
if r is not None:
   print("Valid number")
else:
   print("Not a valid number")


email = input("Enter e-mail address: ")
regex = re.compile(r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$')
match = regex.fullmatch(email)
if match:
   print("Valid email address")
else:
   print('Not a valid email address')
```

**Output:**

```
Enter mobile number:
Valid number
Enter e-mail address:
Valid email address
```

```
Enter mobile number:
Not a valid number
Enter e-mail address:
Not a valid email address
```

## 3. Demonstrate use of List

```
cars = ['Volvo', 'Subaru', 'Skyline', 'Ford']
print(cars)
print(cars[0])
print(cars[-1])
cars[1]='suzuki'
print(cars)
cars.append('Audi')          # adds 'Audi' at the end of the list
print(cars)
cars.insert(2,'BMW')         # adds 'Audi' at the beginning of the list (at index 2)
print(cars)
del cars[0]
print(cars)
cars = ['Volvo', 'Subaru', 'Skyline', 'Ford']
cars.sort()          # arranges the list alphabetically
print(cars)
cars = ['Volvo', 'Subaru', 'Skyline', 'Ford', 'Isuzu']
print(cars[0:3])
```

**OUTPUT:**

```
['Volvo', 'Subaru', 'Skyline', 'Ford']
Volvo
Ford
['Volvo', 'suzuki', 'Skyline', 'Ford']
['Volvo', 'suzuki', 'Skyline', 'Ford', 'Audi']
['Volvo', 'suzuki', 'BMW', 'Skyline', 'Ford', 'Audi']
['suzuki', 'BMW', 'Skyline', 'Ford', 'Audi']
['Ford', 'Skyline', 'Subaru', 'Volvo']
['Volvo', 'Subaru', 'Skyline']
```

## 4. Demonstrate use of Dictionaries

```
country_capitals = {
  "United States": "Washington D.C.",
  "Italy": "Rome",
  "England": "London"
}

# printing the dictionary
print(country_capitals)
print(len(country_capitals))
print(country_capitals["United States"])

# change the value of "Italy" key to "Rome"
country_capitals["Italy"] = "Rome"

# add an item with "Germany" as key and "Berlin" as its value
country_capitals["Germany"] = "Berlin"

# delete item having "United States" key
del country_capitals["United States"]

print(country_capitals)
```

**OUTPUT:**

```
{'United States': 'Washington D.C.', 'Italy': 'Rome', 'England': 'London'}
3
Washington D.C.
{'Italy': 'Rome', 'England': 'London', 'Germany': 'Berlin'}
```

## 5. Create SQLite Database and Perform Operations on Tables.

```
import sqlite3
try:
   con=sqlite3.connect("studentbase.db")
   cursor=con.cursor()

   #create table student
   table="'create table student(name varchar(30),regno VARCHAR(20), course
VARCHAR(20))'"
   cursor.execute(table)
   print(" student table created successfully")

   #insert values into the table student
   cursor.execute("'insert into student values('Ajay','U21SM001','BCA')'")
   cursor.execute("'insert into student values('Vijay','U21SM002','BBA')'")
   cursor.execute("'insert into student values('Sanjay','U21SM003','BCOM')'")
   cursor.execute("'insert into student values('Rajiv','U21SM004','BSC')'")
   print("Inserted successfully")

   #Display the contents of the table
   data=cursor.execute("'select * from student'")
   for i in data:
      print(i)

   print("**below: updated****")

   #update any row ,change name using regno in where clause
   cursor.execute("'update student set name='Ramesh' where name='Ajay';'")
   data=cursor.execute("'select * from student'")
   for i in data:
      print(i)

   print("**below:deleted***")

   #Delete a row
   cursor.execute("'delete from student where regno='U21SM002';'")
   data=cursor.execute("'select * from student'")
   for i in data:
      print(i)

   cursor.close()
   con.commit()
```





```
except sqlite3.Error as err:
   print("Error:",err)
finally:
   if con:
      con.close()
```

**OUTPUT:**

```
student table created successfully
Inserted successfully
('Ajay', 'U21SM001', 'BCA')
('Vijay', 'U21SM002', 'BBA')
('Sanjay', 'U21SM003', 'BCOM')
('Rajiv', 'U21SM004', 'BSC')
**below: updated****
('Ramesh', 'U21SM001', 'BCA')
('Vijay', 'U21SM002', 'BBA')
('Sanjay', 'U21SM003', 'BCOM')
('Rajiv', 'U21SM004', 'BSC')
**below:deleted***
('Ramesh', 'U21SM001', 'BCA')
('Sanjay', 'U21SM003', 'BCOM')
('Rajiv', 'U21SM004', 'BSC')
```

## 6. Create a GUI using Tkinter module

```
import tkinter
window = tkinter.Tk( )
window.title("example GUI")
window.geometry("400x200") # set window size to 400x200 pixels
window.config(bg="Pink") # set window background color to light blue
label = tkinter.Label(window, text="HI CIT!!")
label.pack( )
window.mainloop( )
```

**Output:**

## 7. Demonstrate Exceptions in Python

```
try:
   numerator = 10
   denominator = 0

   result = numerator/denominator

   print(result)
except:
   print("Error: Denominator cannot be 0.")

finally:
   print("This is finally block.")
```

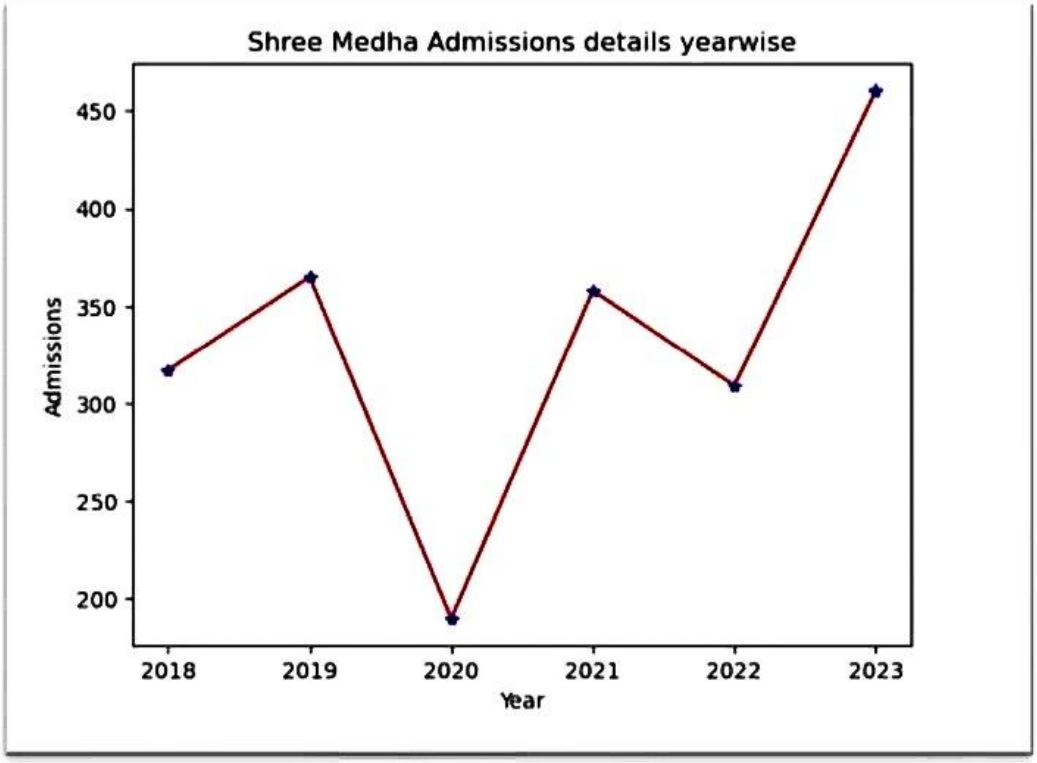
**OUTPUT:**

```
Error: Denominator cannot be 0.
This is finally block.
```

## 8. Drawing Line chart and Bar chart using Matplotlib

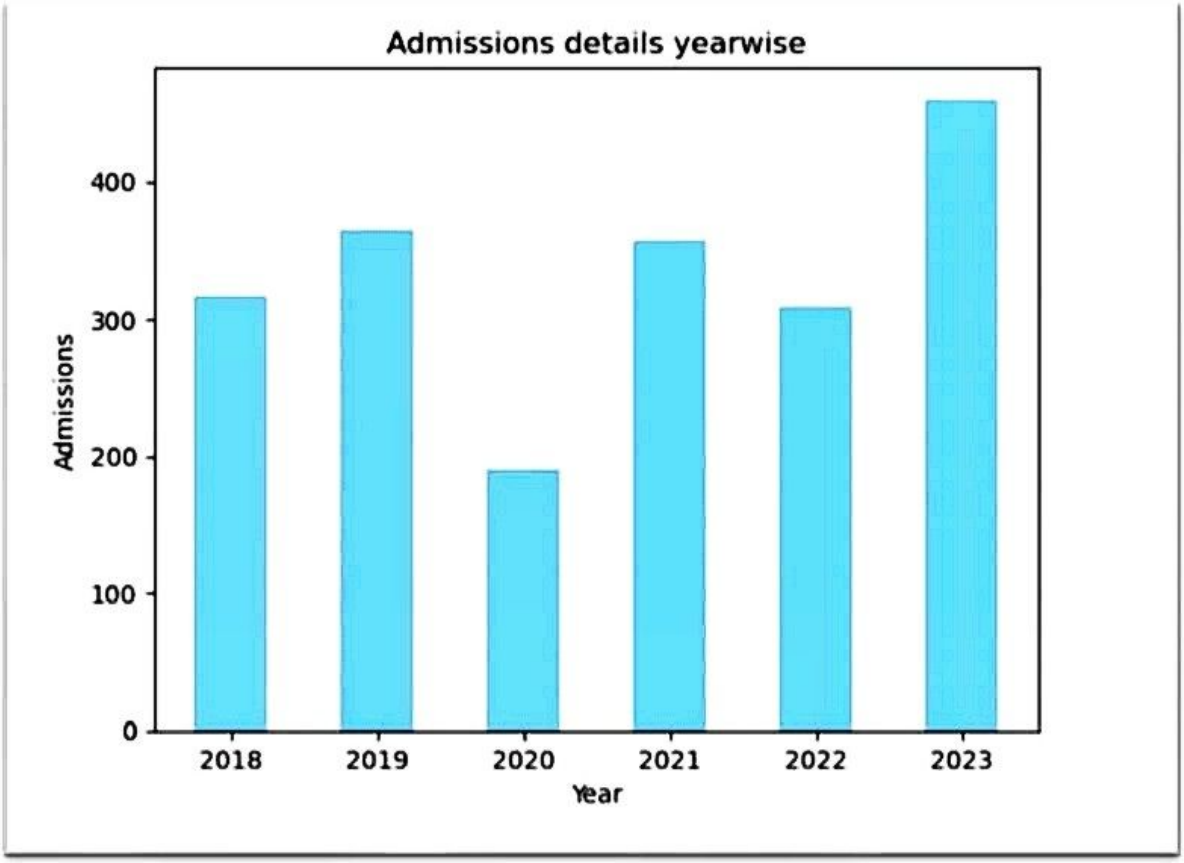
```
#line chart
import matplotlib.pyplot as plt
year = [2018, 2019, 2020, 2021, 2022, 2023]
admissions = [317, 365, 190, 358, 309, 460]
plt.plot(year,
admissions,linestyle="solid",linewidth=1.5,color='red',marker="*",mec='b')
plt.title("Shree Medha Admissions details yearwise")
plt.xlabel('Year')
plt.ylabel('Admissions')
plt.show()
```

**OUTPUT:**



```
#bar chart
import matplotlib.pyplot as plt
year = [2018, 2019, 2020, 2021, 2022, 2023]
admissions = [317, 365, 190, 358, 309, 460]
plt.bar(year, admissions,width=0.5,color="skyblue")
plt.title("Admissions details yearwise")
plt.xlabel('Year')
plt.ylabel('Admissions')
plt.show()
```

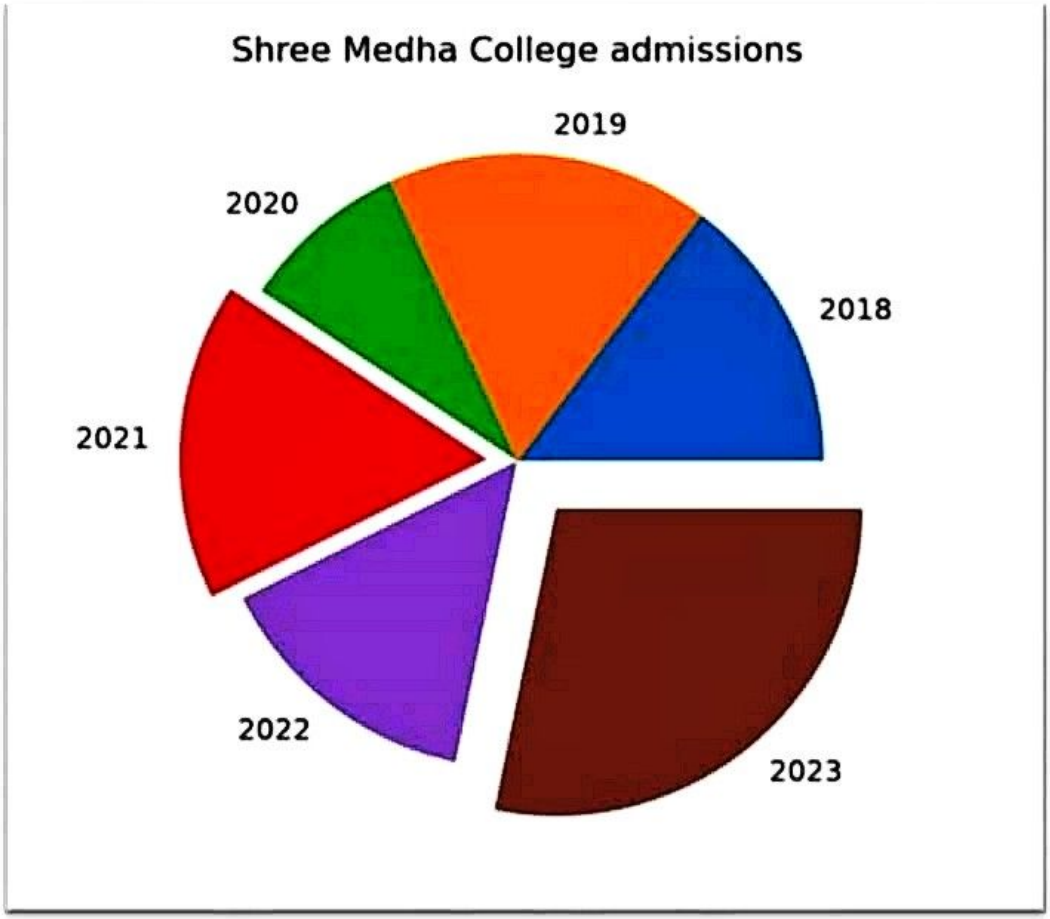
**OUTPUT:**



## 9. Drawing Histogram and Pie chart using Matplotlib

```
import matplotlib.pyplot as plt
years = [2018, 2019, 2020, 2021, 2022, 2023]
admissions = [317, 365, 190, 358, 309, 605]
plt.title("Shree Medha College admissions")
plt.pie(admissions, labels=years, explode=[0, 0, 0, 0.1, 0, 0.2])
plt.hist(admissions)
plt.show()
```

**OUTPUT:**

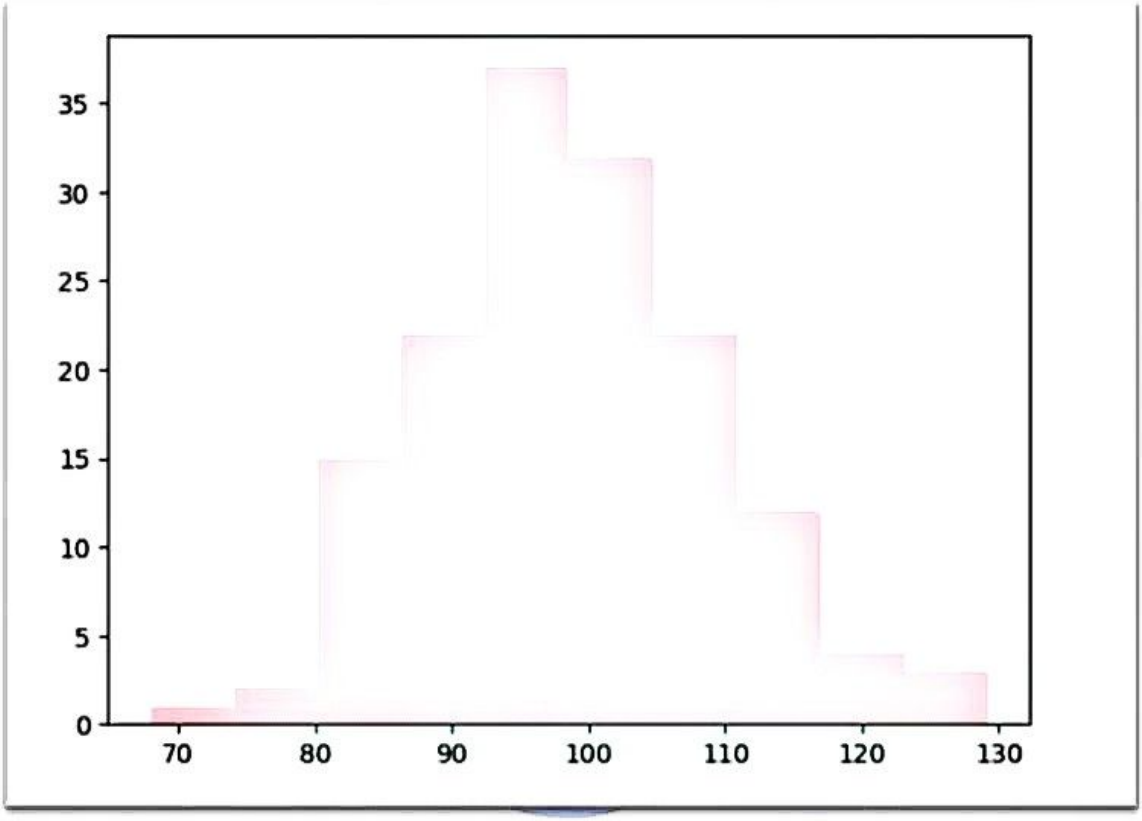


```
#Drawing Histogram chart using Matplotlib.
import matplotlib.pyplot as plt
import numpy as np

x = np.random.normal(100, 10, 150)

plt.hist(x,color='pink')
plt.show()
plt.show()
```

**OUTPUT:**



## 10. Create Array using NumPy and Perform Operations on Array.

```
import pandas as pd
import numpy as np
a = np.array([10, 20, 30, 40])
print(a)
print("******")
b=np.array([10, 20],[30, 40])
print(b)
print("******")
for i in range(len(a)):
   print(a[i])
print("********")
for i in range(len(b)):
   for j in range(len(b)):
      print(b[i][j])
```

**OUTPUT:**

```
[10 20 30 40]
*****
[[10 20]
 [30 40]]
*****
10
20
30
40
********
10
20
30
40
```

## 11. Create DataFrame from Excel sheet using Pandas and Perform Operations on DataFrames

```
import pandas as pd
print("--reading data from excel using data frames----")
df1=pd.read_excel("students.xlsx",na_values="NA")
print(df1)
cols=[0,1,3]
print("--accessing required columns---")
df2=pd.read_excel("students.xlsx",na_values="NA",usecols='A')
print(df2)
print()
print("--accessing specific sheet---")
df3=pd.read_excel("students.xlsx",na_values="NA",sheet_name='Sheet2')
print(df3)
print()
print("--accessing all the sheets ---")
print()
df4=pd.read_excel("students.xlsx",na_values="NA",sheet_name=None)
print()
print(df4)
```

**OUTPUT:**

```
--Reading data from excel using data frames----
```

| | Name | English | Matha | Kannada |
|---|---|---|---|---|
| 0 | Asha | 15 | 20 | 18 |
| 1 | Vijay | 20 | 12 | 19 |
| 2 | Sanjay | 16 | 13 | 17 |
| 3 | Rajiv | 18 | 16 | 16 |
| 4 | Anup | 17 | 18 | 15 |



```
--Accessing Required Columns---
```

| | Name |
|---|---|
| 0 | Asha |
| 1 | Vijay |
| 2 | Sanjay |
| 3 | Rajiv |
| 4 | Anup |

```
--Accessing specific sheet---
```

| | Name | Python |
|---|---|---|
| 0 | Asha | 15 |

| | | |
|---|---|---|
| 1 | Vijay | 20 |
| 2 | Sanjay | 16 |
| 3 | Rajiv | 18 |

--Accessing all the sheets ---

| {'Sheet1': | Name | English | Matha | Kannada |
|---|---|---|---|---|
| 0 | Asha | 15 | 20 | 18 |
| 1 | Vijay | 20 | 12 | 19 |
| 2 | Sanjay | 16 | 13 | 17 |
| 3 | Rajiv | 18 | 16 | 16 |
| 4 | Anup | 17 | 18 | 15 |

| 'Sheet2': | Name | Python |
|---|---|---|
| 0 | Asha | 15 |
| 1 | Vijay | 20 |
| 2 | Sanjay | 16 |
| 3 | Rajiv | 18 |

'Sheet3': Empty DataFrame

Columns: []

Index: []}