Page No.
Date / /

program : 72 : Write a program OF private method :

```
class Car :

    def __init__(self):

        self.__updateSoftware()

    def drive(self):
        print("Driving")

    def __updateSoftware(self):
        print("Updating Software")

redcar = Car()
redcar.drive()

#redcar.__updateSoftware  it will give error because method not call directly
```

Output :

```
Updating Software
Driving
```

Page No.
Date / /

Program: 80: Write a program OF Exception:

```
try:
    X = int(input(" Enter Number1 : "))
    Y = int(input(" Enter Number2 : "))

    print(x / y)

except zeroDivisionError:

        print(" program successfully complete")

Finally:

    print (" End program")
    print(" End ... End ... End ")
```

Output:

```
Enter Number1 : 10
Enter Number2 : 5
2.0
program successfully complete
End program
End ... End ... End
```

- ```
  Enter Number1 : 10
  Enter Number2 : 0
  End program
  End ... End ... End
  ZeroDivisionError: division by Error
  ```
  ```

Page No.
Date / /

program 81: Write a program of to handle ZeroDivisionError Example:

```
#An exception handling Example

try:
    F=open("myFile","w")

    a,b=[int(x) for x in input("Enter two
            numbers:").split()]

    c=a/b
    F.write("writing %d into myFile"%c)

except ZeroDivisonError:
        print("Divison Error")
        F.write("Divison by zero happend")

Finally:
    F.close()
    print("File closed")
```

Output:

```
Enter two numbers: 10 5
File closed
```

myFile:- writing 2 into myFile

```
Enter two numbers: 10 0
File closed
Error:- - -
```

Page No.
Date / /

Program: 82: Write a program OF to handle Syntax Error given by eval() function.

```
# Example of SyntaxError

try:
    date = eval(input("Enter date: "))

except SyntaxError:
    print("Invalid date entered")

else:
    print("you entered:", date)
```

Output:

```
Enter date: 2020, 10, 3

you entered: (2020, 10, 3)


- Enter date: 2016, 106, 3

Invalid date entered
```

Page No.
Date / /

Program: 83: Write a program OF using the assert statement and catching AssertionError:

#Handling AssertionError

```
try:
    x = int(input("Enter a number between 5 and 10:"))
    assert x >= 5 and x <= 10
    print("The number entered:", x)
except AssertionError:
    print("The condition is not fulfilled")
```

Output:

Enter a number between 5 and 10: 11

The condition is not fulfilled

- Enter a number between 5 and 10: 8

The number entered: 8

Searching Techniques

Page No.
Date / /

program: 84: write a program of Linear Search:

```
#Linear Search Example

def linear_search(values, search_for):

        search_at = 0
        search_res = False
        #match the value with each data
                                element
        while search_at < len(values)
                and search_res is False:

            if values[search_at] ==
                                search_for:
                search_res = True
            else:
                search_at = search_at +
                                1
        return search_res

s = int(input("Enter element that you want
                        to search:"))
list1 = [10, 20, 30, 40, 50, 60]
ans = linear_search(list1, s)
if (ans == True):
    print("Element Found")
else:
    print("Element not Found")
```

Page No.
Date / /

Output :

Enter element that you want to search : 20

Element Found

- Enter element that you want to search : 5

Element not Found

Page No.
Date / /

Program: 15: Write a program of binary search:

```
# Iterative binary search function
# It returns index of x in given array
                    arr if present,
# else return -1

def binary_search(arr, x):

    low = 0
    high = len(arr) - 1
    mid = 0
    while low <= high:

        mid = (high + low) // 2
        # If x is greater than left half
        if arr[mid] < x:
            low = mid + 1

        # If x is smaller, ignore right
                            half
        elif arr[mid] > x:
            high = mid - 1

        # means x is present at mid
        else:
            return mid
    # If we reach here, then the element
                    was not present
    return -1
```

Page No.
Date / /

Test array

```
arr = [2, 3, 4, 20, 40]
x = 40 // or x = 33
# Function call

result = binary_search(arr, x)
if result != -1:
    print("Element is present at index",
                                str(result))
else:
    print("Element is not present in
                                array")
```

Output:

Element is present at index 4

IF x = 33, 50

Element is not present in array

Page No.
Date / /

Program: 16: Write a program of Selection Sort:

```
#python program for implementation of
                    selection sort
import sys

A = [50, 20, 30, 10, 40, 60]
#traverse through all array element
for i in range(len(A)):

    #Find the minimum element in
                        remaining
    #unsorted array

    min_idx = i

    for j in range(i+1, len(A)):

        if A[min_idx] > A[j]:

            min_idx = j
    #swap the found minimum element
            with the first element

    A[i], A[min_idx] = A[min_idx],
                        A[i]
#Driver code to test above
print("Sorted array")
```

Page No.
Date / /

```
for i in range(len(A)):
    print("%d" %A[i])
```

Output:

```
Sorted array
10
20
30
40
50
60
```

Page No.
Date / /

Program: 81: write a program OF Bubble sort.

```
def bubble_sort(arr):

    def swap(i, j):

        arr[i], arr[j] = arr[j], arr[i]
    n = len(arr)
    swapped = True
    x = -1

    while swapped:

        swapped = False
        x = x+1

        for i in range(1, n-x):

            if arr[i-1] > arr[i]:
                swap(i-1, i)
                swapped = True

    print(arr)

bubble_sort([1, 2, 6, 3, 4, 5])
```

Output:

[1, 2, 3, 4, 5, 6]

Page No.
Date / /

Program 8: Write a program OF Hashtables:

```
#Declare a dictionary

dict = {'Name': 'zara', 'Age': 7, 'class':
                                    'First'}

#Accessing the dictionary with its key

print ("dict['Name']: ", dict['Name'])
print ("dict['Age']: ", dict['Age'])
```

output:

```
dict['Name']: zara

dict['Age']  : 7
```

Plotting Using Python

Page No.
Date / /

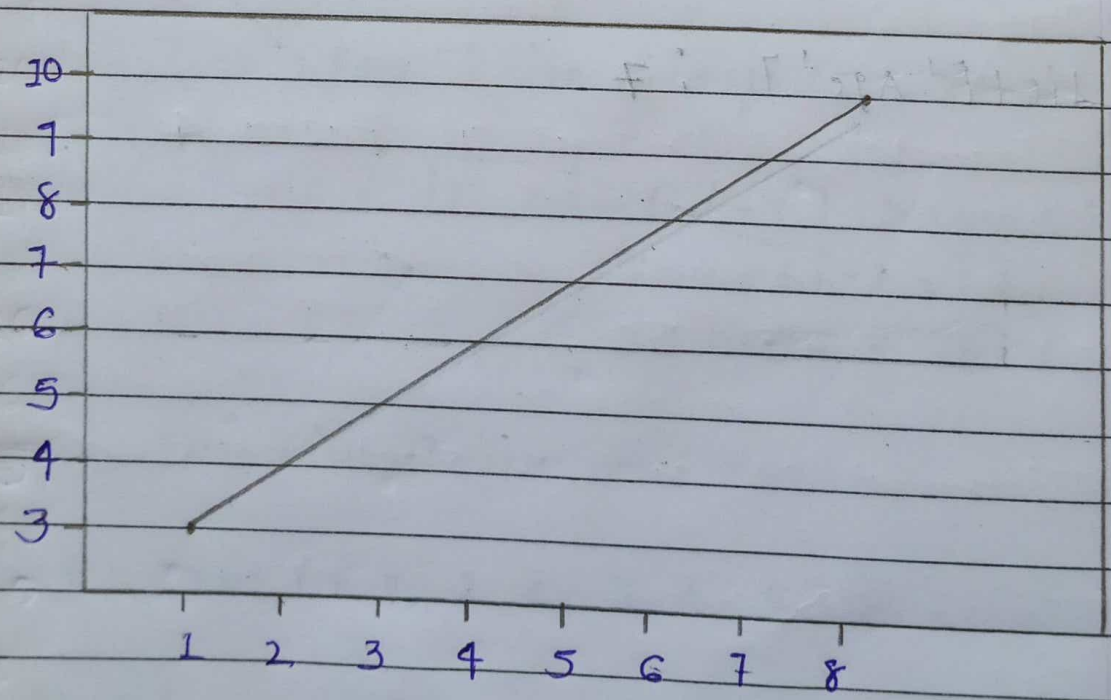
program: 89: write a program OF Draw a line in diagram From (1, 3) to position (8, 10):

```
import matplotlib.pyplot as plt
import numpy as np

Xpoints = np.array([1, 8])
Ypoints = np.array([3, 10])

plt.plot(xpoints, ypoints)
plt.show()
```

output:



Page No.
Date / /

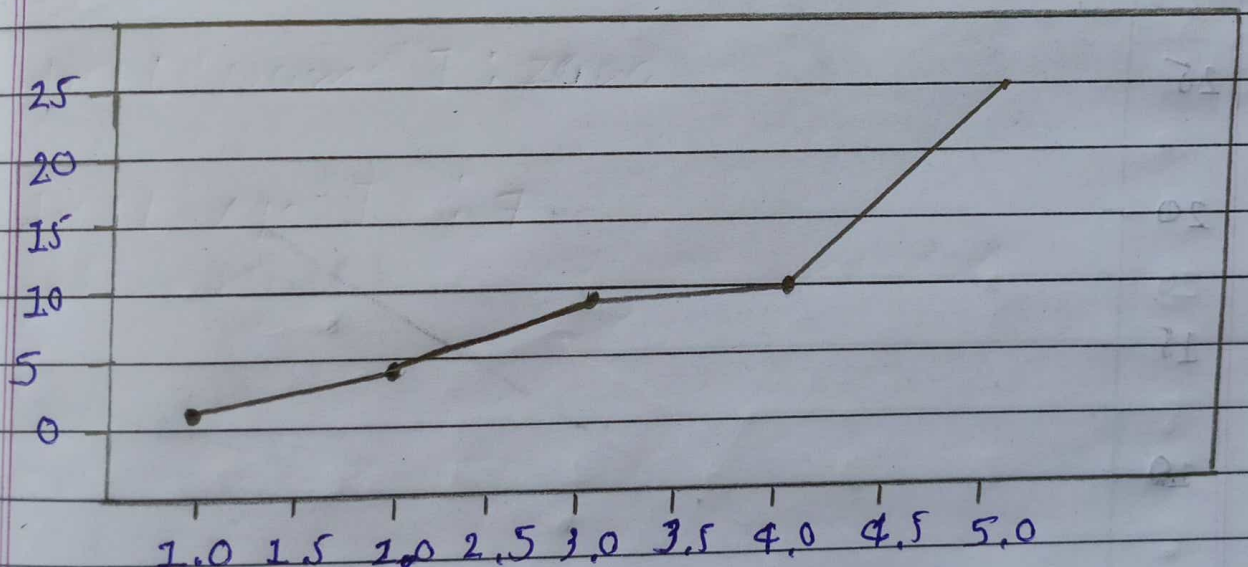
program: 90: Draw a line in diagram without Array:

```
import numpy as np
import pylab as pl

x = [1, 2, 3, 4, 5]
y = [1, 4, 9, 10, 25]

pl.plot(x, y)
pl.show()
```

Output:



Page No.
Date / /

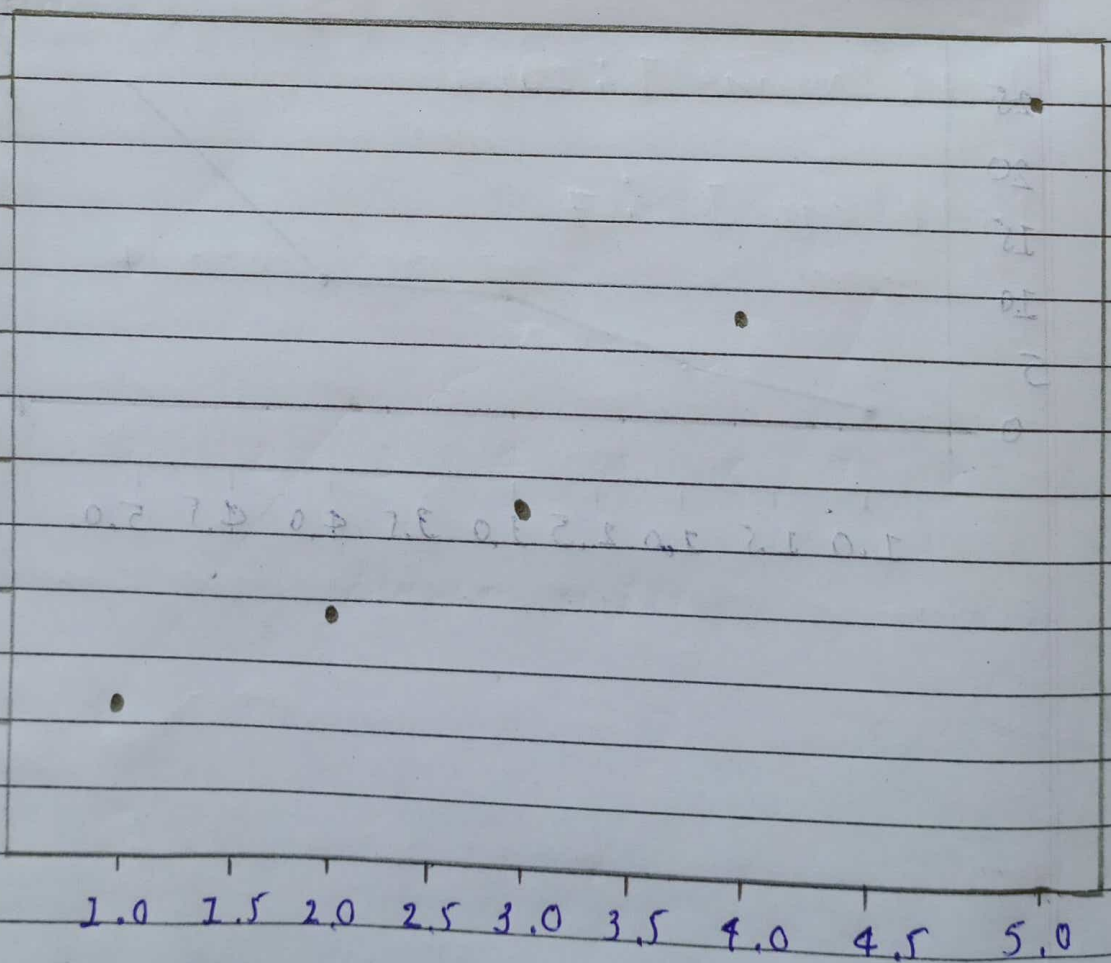
Program: 9: Write a program OF to demonstrate Scatter plot:

```
import numpy as np
import pylab as pl

x = [1, 2, 3, 4, 5]
y = [1, 4, 9, 16, 25]

pl.scatter(x, y)  OR  pl.plot(x, y, 'ro')
pl.show()
```

Output:



Page No.
Date / /

Program: 92: Write a program to Changing the Line Color and Style:

```
import matplotlib.pyplot as plt
import numpy as np

y1 = np.array([3,8,1,10])

y2 = np.array([6,2,7,11])

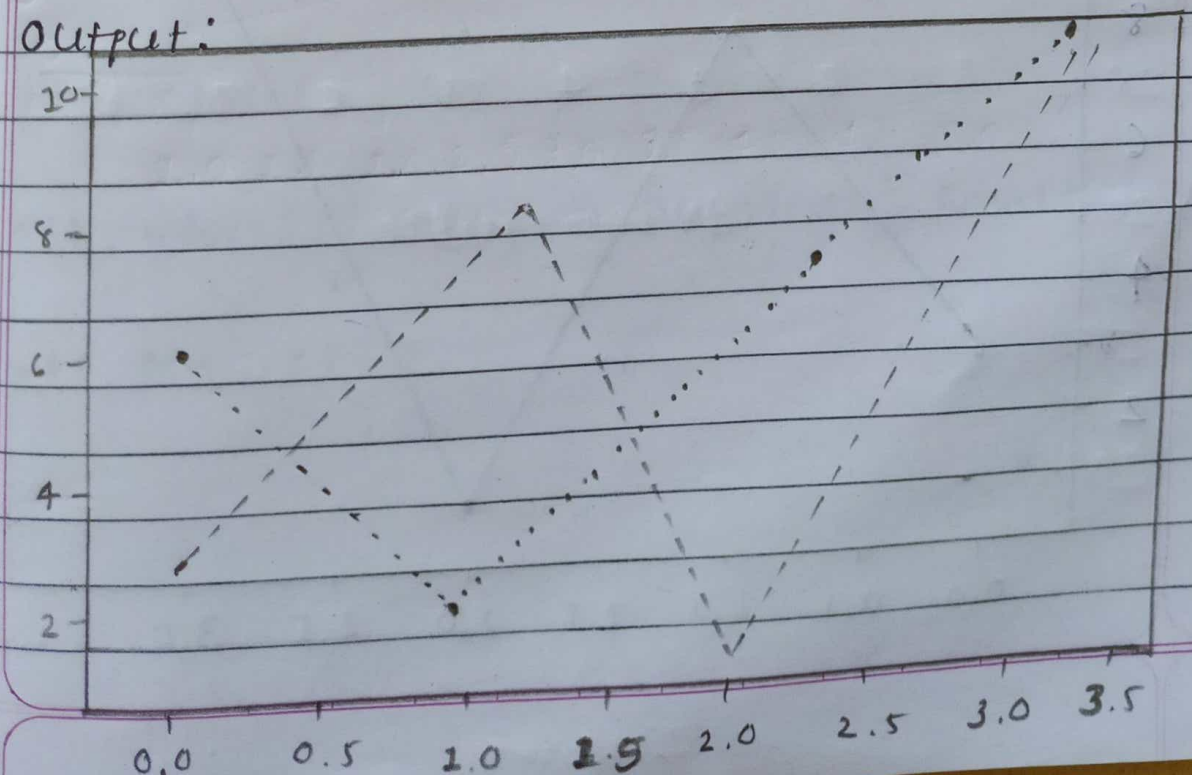
plt.plot(y1, color='r', linestyle='--')

plt.plot(y2, color='g', linestyle=':')

# plt.plot(y1, c='r', ls='dotted', line-
                        width='5')

plt.show()
```

Output:



Page No.
Date / /

program: 93: Write a program to demonstrate marker:

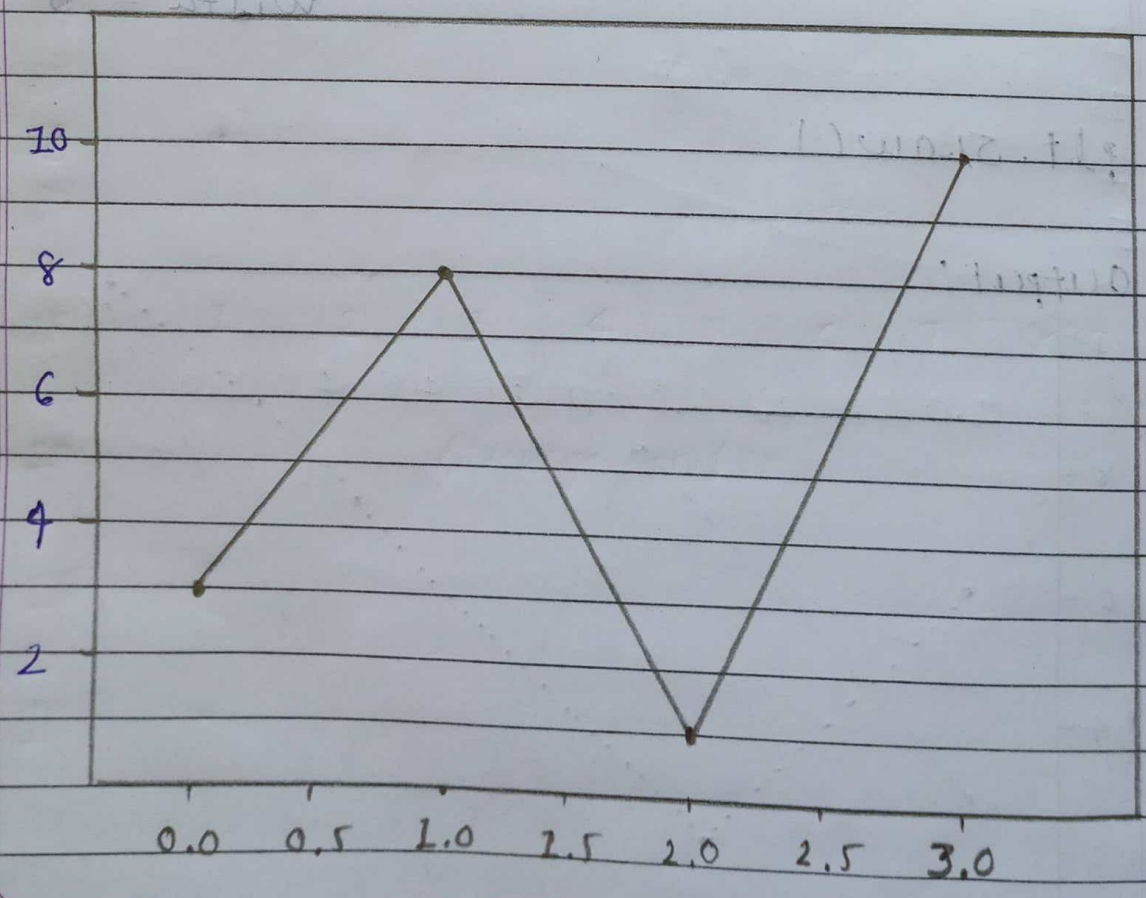
```
import matplotlib.pyplot as plt
import numpy as np

ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, marker = 'o', ms = 10,
         mec = 'r', mfc = 'g')

plt.show()
```

output:



Page No.
Date / /

program: 94: write a program to demonstrate labels, Title & Font properties:

```
import matplotlib.pyplot as plt
import numpy as np

xpoints = np.array([1, 8])

ypoints = np.array([3, 10])

Font1 = {'family': 'serif', 'color': 'red', 'size': 20}

Font2 = {'family': 'Roman', 'color': 'Darkred', 'size': 15}

plt.plot(xpoints, ypoints)

plt.title("Sports watch Data", fontdict=Font1)

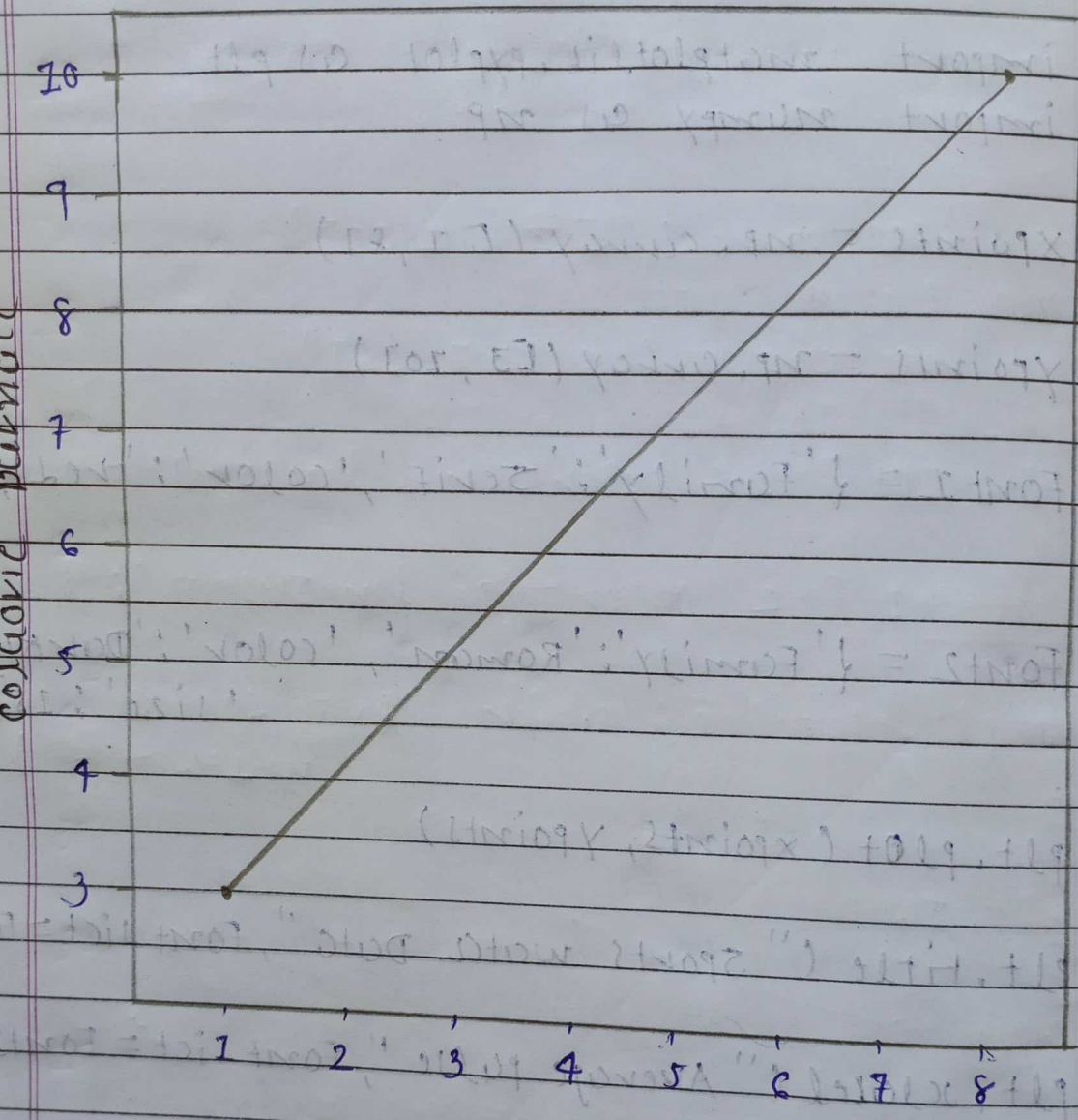
plt.xlabel("Average pulse", fontdict=Font2)

plt.ylabel("Calloric Burnase", fontdict=Font1)

plt.show()
```

Page No.
Date / /

Output: Sports watch Data



Page No.
Date / /

program:95: write a program of grid:

```
import matplotlib.pyplot as plt
import numpy as np

#plot 1

x = np.array([0,1,2,3])
y = np.array([3,8,1,10])

plt.subplot(1,2,1)
plt.plot(x,y '*:r')
plt.title("sales")
plt.grid()

#plot 2

x = np.array([0,1,2,3])
y = np.array([10,20,30,40])

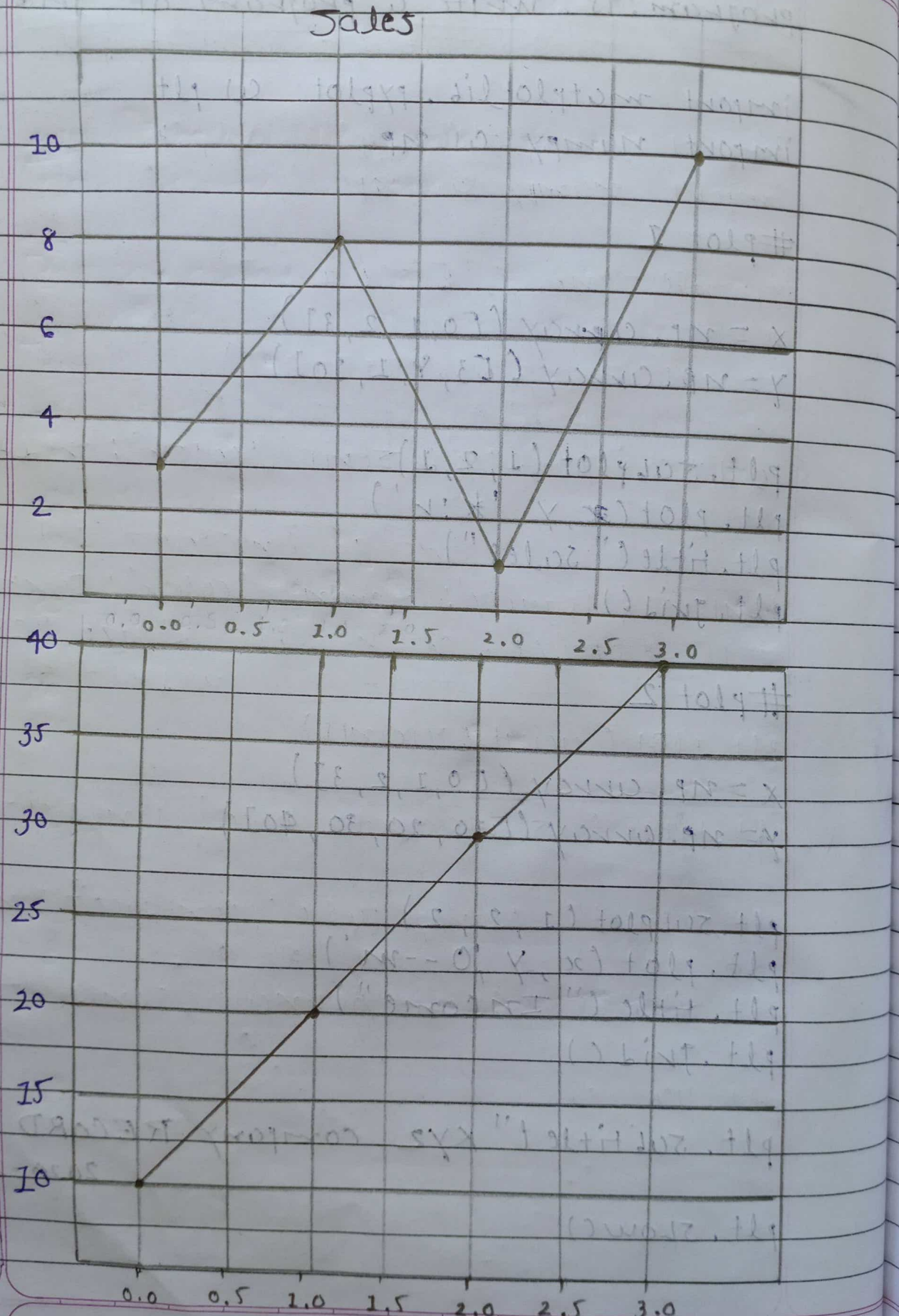
plt.subplot(1,2,2)
plt.plot(x,y,'O--m')
plt.title("Income")
plt.grid()

plt.suptitle("XYZ company RECORD
                    2020-21")

plt.show()
```

Page No.
Date / /

XYZ Company RECORD 2020-21



Page No.
Date / /

program : 96 : write a program of

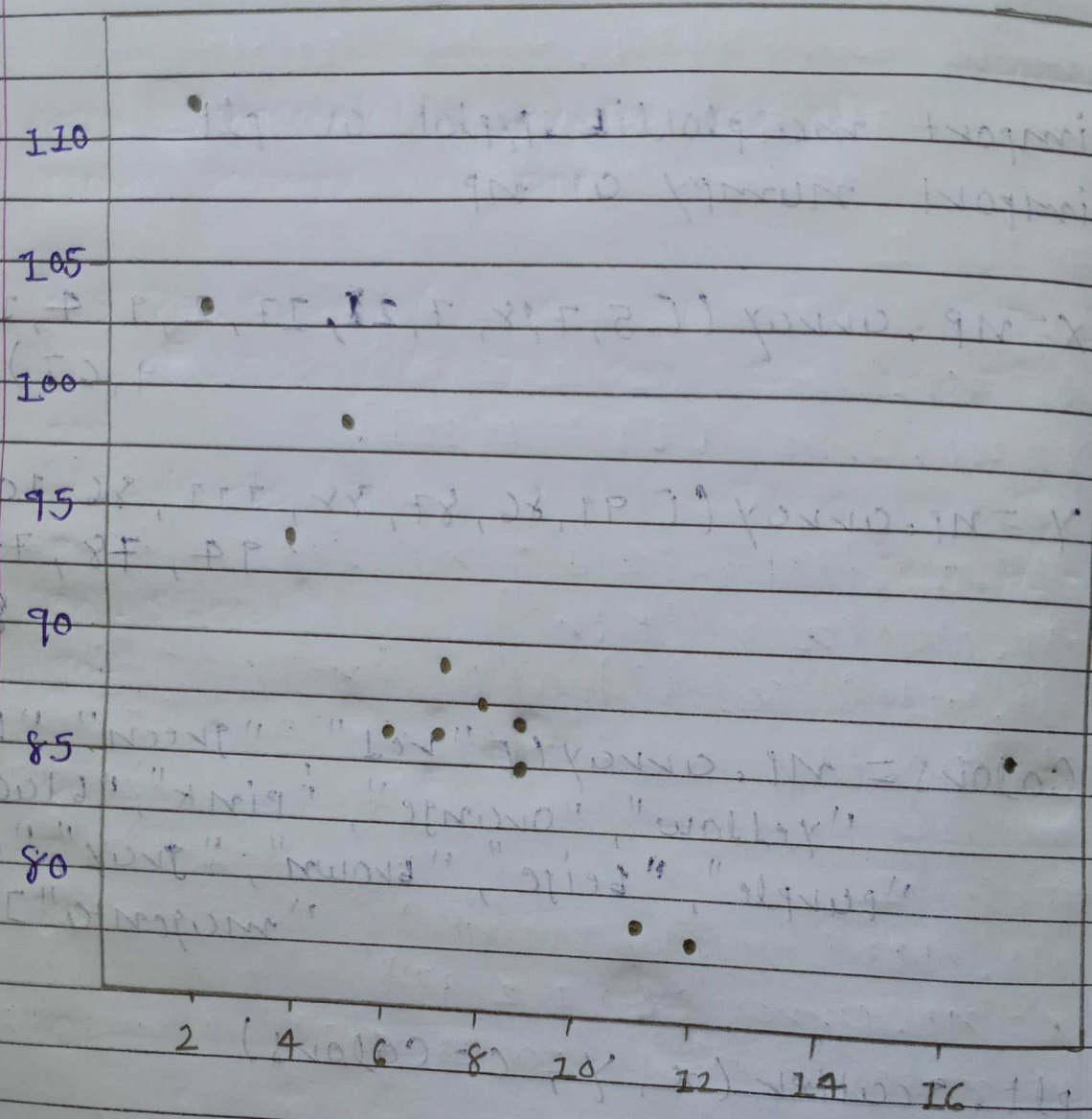
```
import matplotlib.pyplot as plt
import numpy as np

x = np.array([5, 7, 8, 7, 2, 17, 2, 9, 4, 11, 12,
              9, 6])

y = np.array([99, 86, 87, 88, 111, 86, 103, 87,
              94, 78, 77, 85,
              86])

colors = np.array(["red", "green", "blue",
        "yellow", "orange", "pink", "black",
    "purple", "beige", "brown", "gray", "cyan",
                   "magenta"])

plt.scatter(x, y, c = colors)
plt.show()
```


110
105
100
95
90
85
80
2
4
6
8
10
12
14
16

Page No.
Date / /

program: 97:

```
import matplotlib.pyplot as plt
import numpy as np

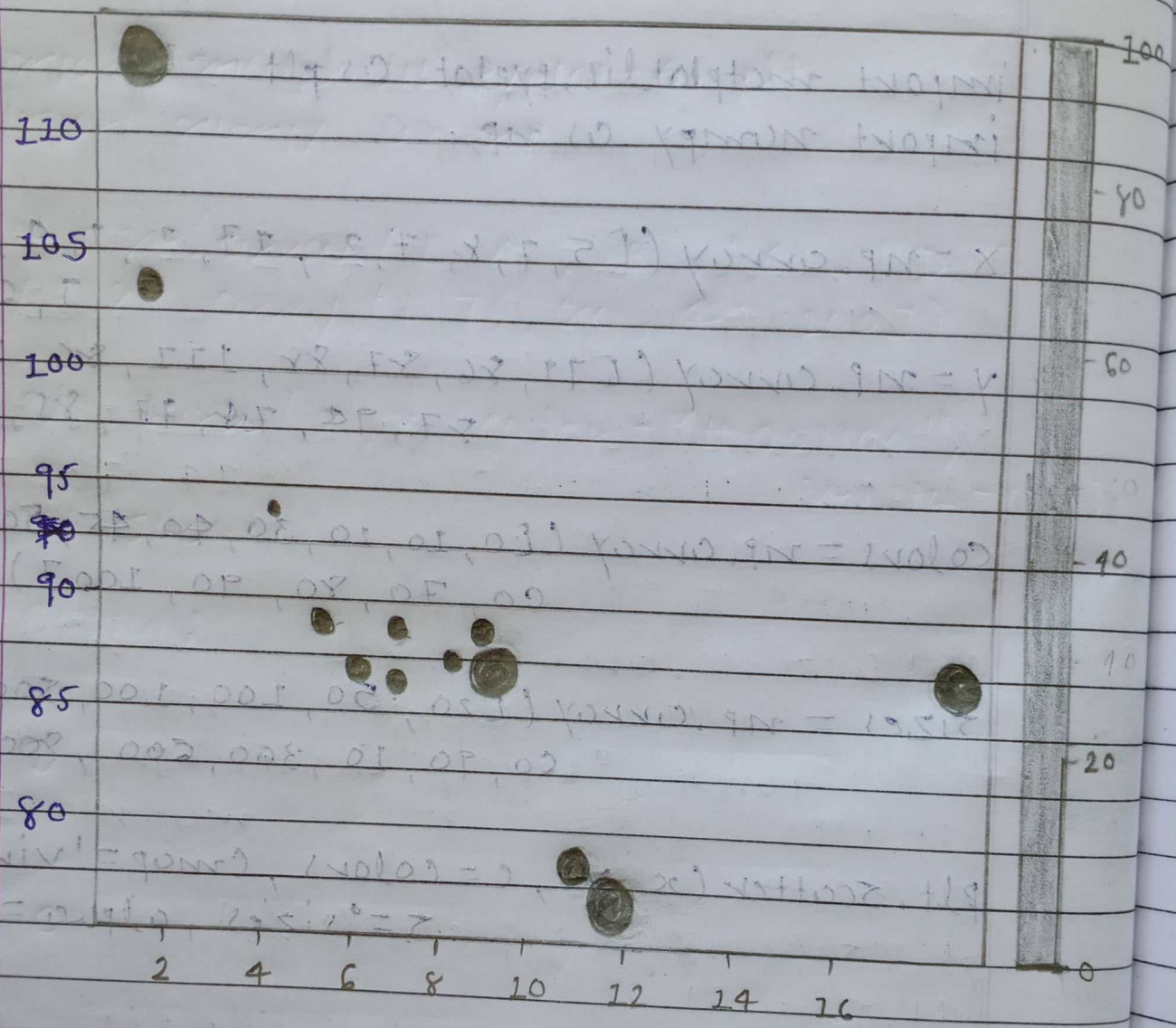
x = np.array([5, 7, 8, 7, 2, 17, 2, 9, 4, 11, 12,
              9, 6])
y = np.array([99, 86, 87, 88, 111, 86, 103,
              87, 94, 78, 77, 85, 86])

colors = np.array([0, 10, 20, 30, 40, 45, 50, 55,
                   60, 70, 80, 90, 100])

sizes = np.array([20, 50, 100, 100, 500, 1000,
                  60, 90, 10, 300, 600, 800, 75])

plt.scatter(x, y, c=colors, cmap='viridis',
            s=sizes, alpha=0.5)

plt.colorbar()
plt.show()
```


110
105
100
95
90
85
80
2
4
6
8
10
12
14
16
100
80
60
40
20
0

Page No.
Date / /

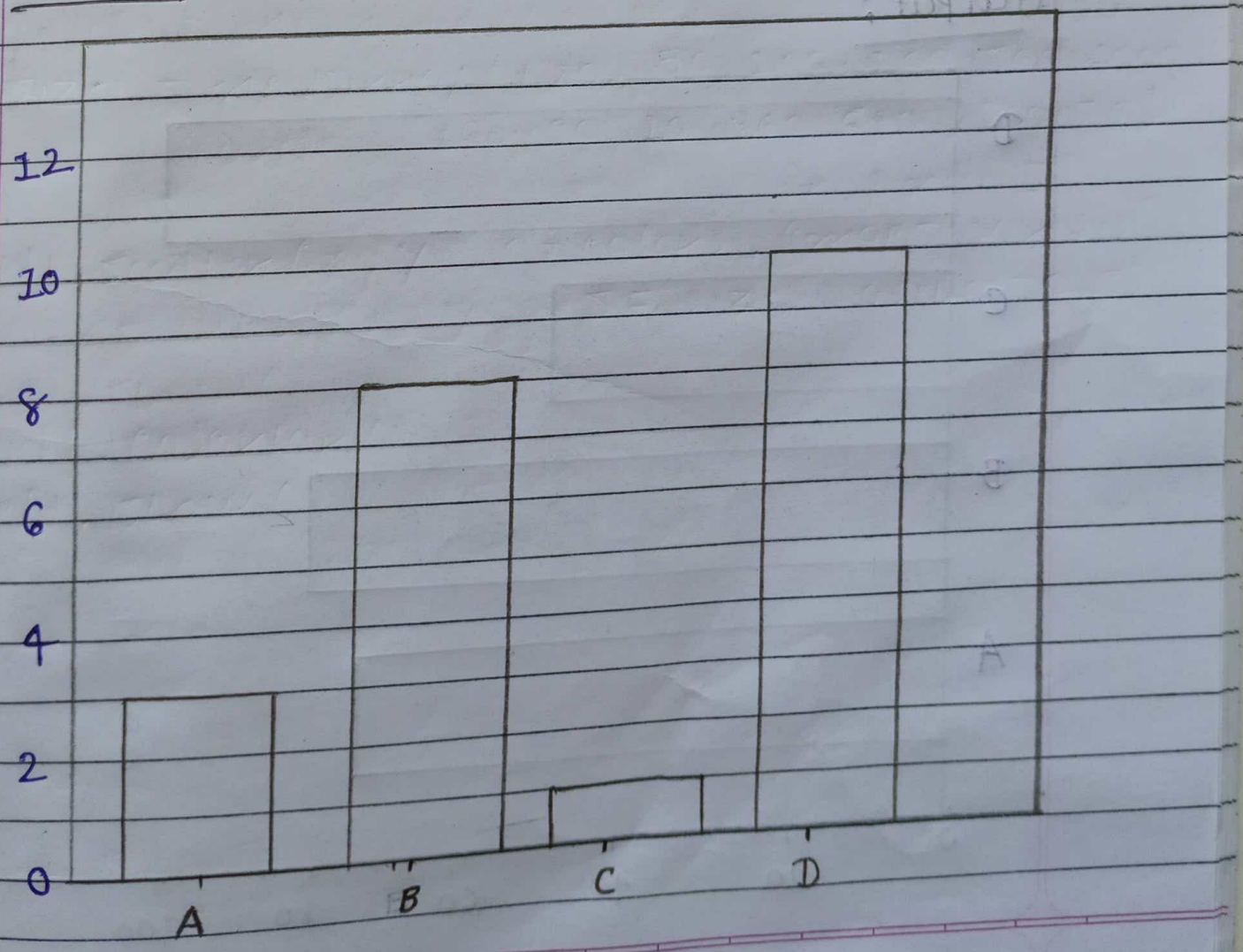
program :98 : Draw 4 bars vertical :

```
import matplotlib.pyplot as plt
import numpy as np

x = np.array(["A", "B", "C", "D"])
y = np.array([3, 8, 1, 10])

plt.bar(x, y)
plt.show()
```

output :



Page No.
Date / /

program:77: Draws horizontal bars:

```
import matplotlib.pyplot as plt
import numpy as np

x = np.array(["A","B","C","D"])
y = np.array([75, 80, 45, 100])

plt.barh(x, y, color="#009900",
                    width=0.5)
plt.show()
```

Output: