INSTITUTION: KCA University

NAME: Raphael Kang’eri

UNIT NAME: Programming for Data Science

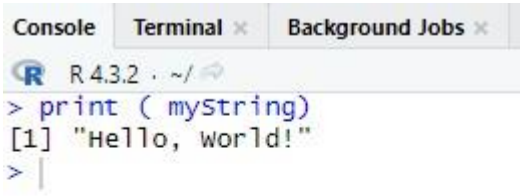
UNIT CODE: BSD 320

INTRODUCTION TO R FOR DATA SCIENCE

1. # My first program in R Programming

   myString <- "Hello, World!"

   print (myString)



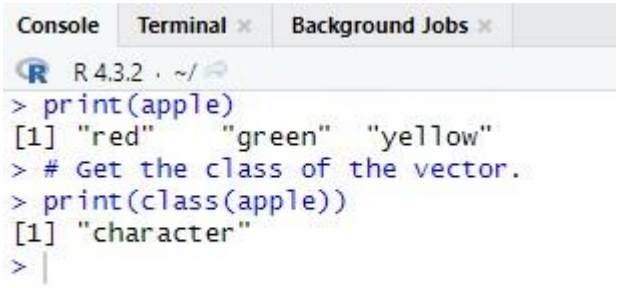
2. # Create a vector.

   apple <- c('red','green',"yellow")

   print(apple)

   # Get the class of the vector.

   print(class(apple))

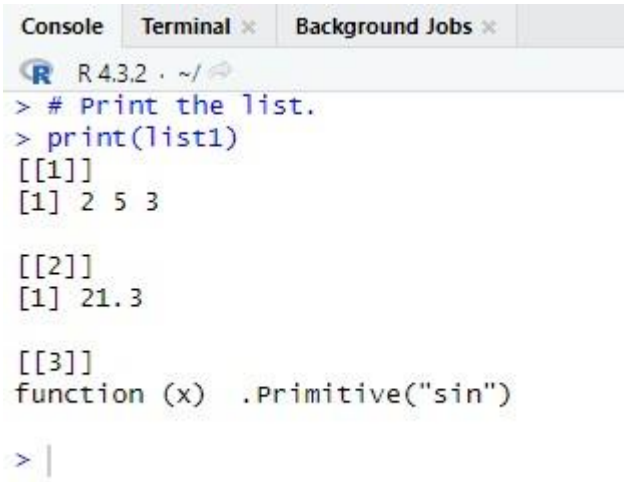


3. # Create a list.

   list1 <- list(c(2,5,3),21.3,sin)

   # Print the list.

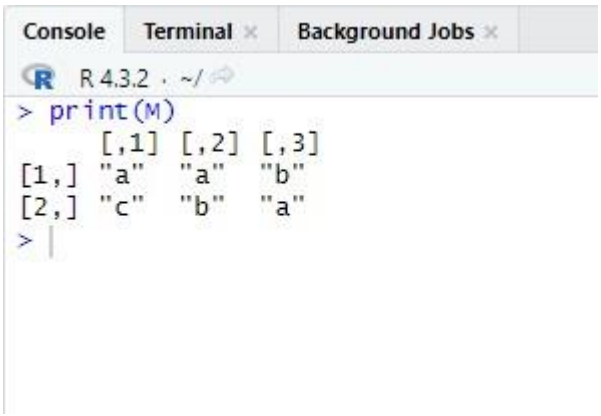
   print(list1)



4. # Create a matrix.

   M = matrix(c('a','a','b','c','b','a'), nrow = 2, ncol = 3, byrow = TRUE)

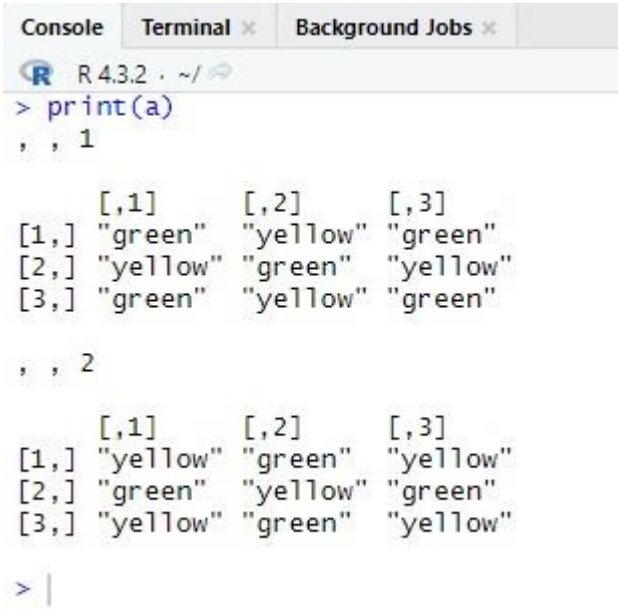
   print(M)



5. # Create an array.

   a <- array(c('green','yellow'),dim = c(3,3,2))

   print(a)



6. # Create a vector.

   apple_colors <- c('green','green','yellow','red','red','red','green')

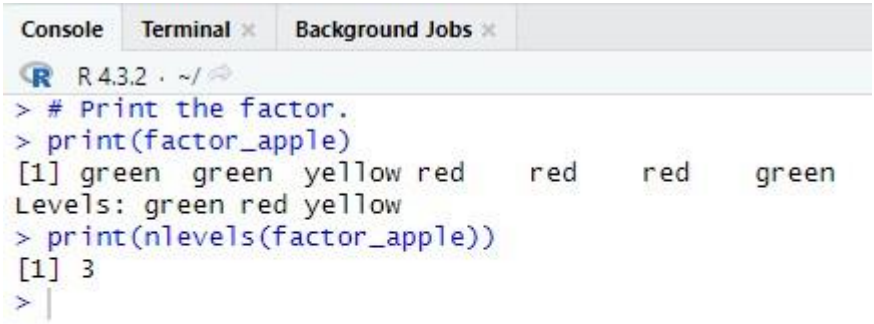
   # Create a factor object.

   factor_apple <- factor(apple_colors)

   # Print the factor.

   print(factor_apple)

   print(nlevels(factor_apple))



7. # Create the data frame.

   BMI <- data.frame(

   gender = c("Male", "Male","Female"),

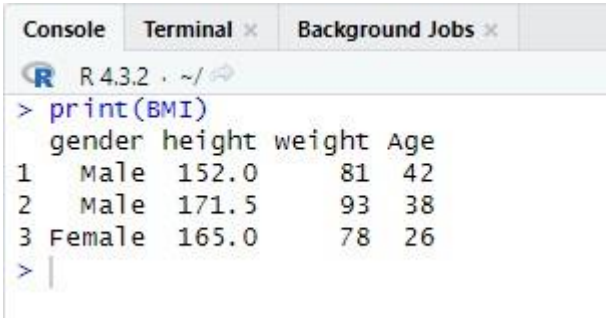
   height = c(152, 171.5, 165),

   weight = c(81,93, 78),

   Age = c(42,38,26)

)

   print(BMI)



8. # Assignment using equal operator.

   var.1 = c(0,1,2,3)

   # Assignment using leftward operator.

   var.2 <- c("learn","R")

   # Assignment using rightward operator.

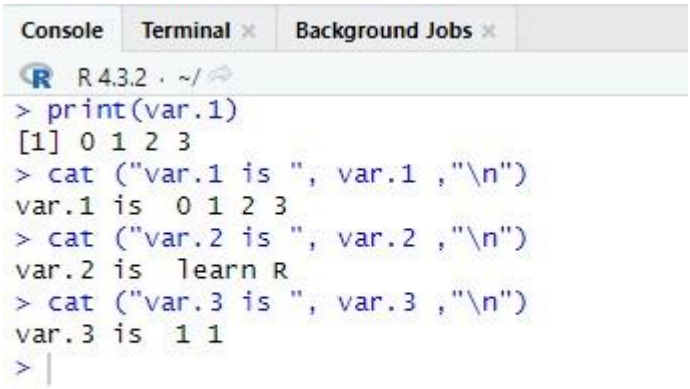
   c(TRUE,1) -> var.3

   print(var.1)

   cat ("var.1 is ", var.1 ,"\n")

   cat ("var.2 is ", var.2 ,"\n")

   cat ("var.3 is ", var.3 ,"\n")



9. var_x <- "Hello"

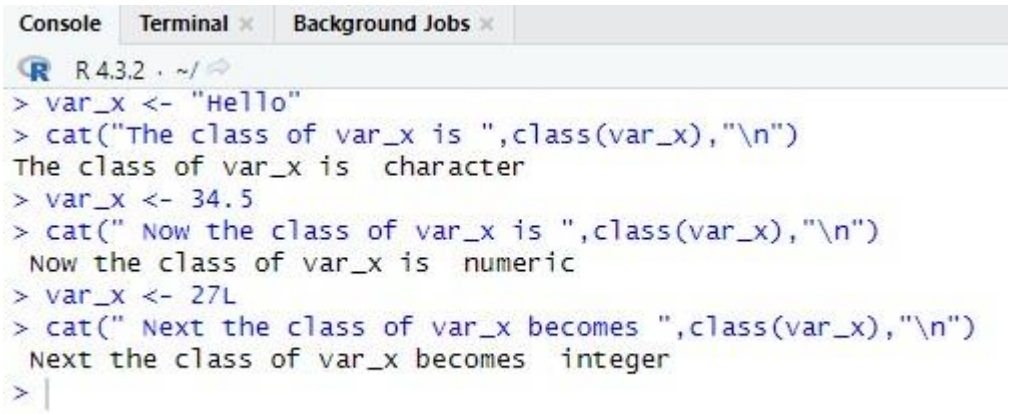
   cat("The class of var_x is ",class(var_x),"\n")

   var_x <- 34.5

   cat(" Now the class of var_x is ",class(var_x),"\n")

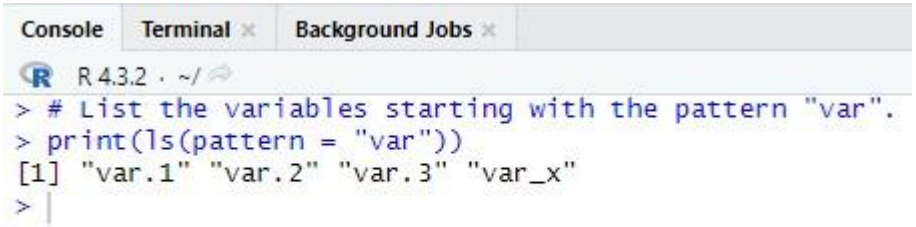
   var_x <- 27L

   cat(" Next the class of var_x becomes ",class(var_x),"\n")

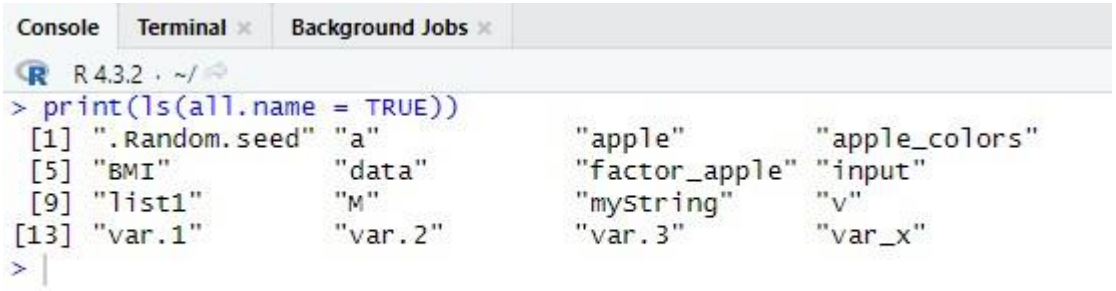


10. # List the variables starting with the pattern "var".

    print(ls(pattern = "var"))



11. print(ls(all.name = TRUE))



1.

INSTITUTION: KCA University

NAME: Raphael Kang’eri

UNIT NAME: Programming for Data Science

UNIT CODE: BSD 320

R-CSV

1. # Get and print current working directory.

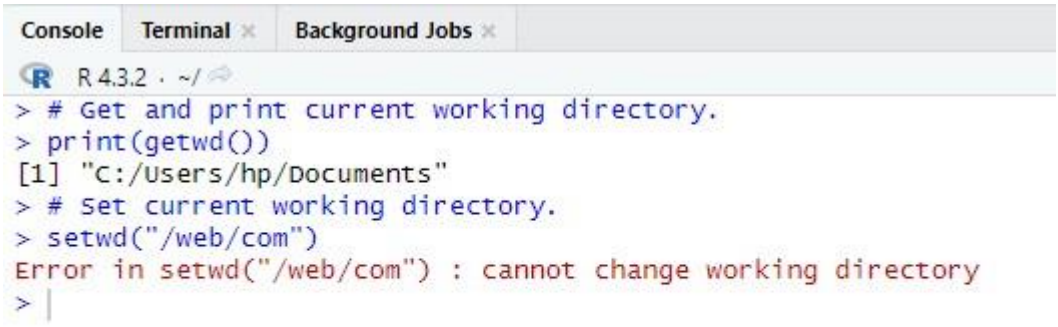
   print(getwd())

   # Set current working directory.

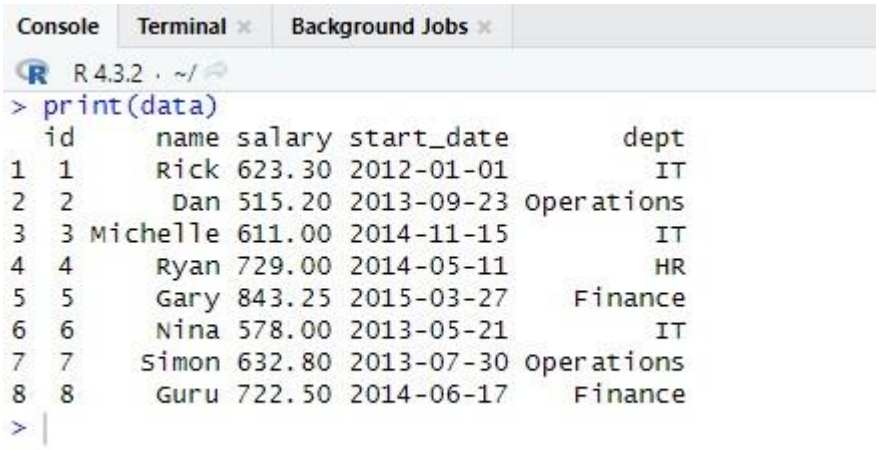
   setwd("/web/com")

   # Get and print current working directory.

   print(getwd())



2. data <- read.csv("input.csv") print(data)

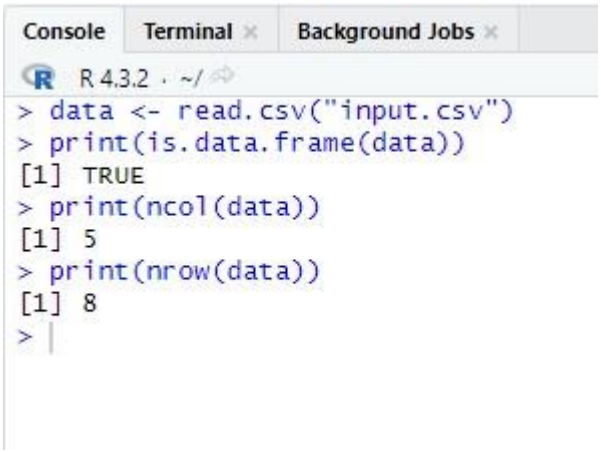


3. data <- read.csv("input.csv")

   print(is.data.frame(data))

   print(ncol(data))

   print(nrow(data))



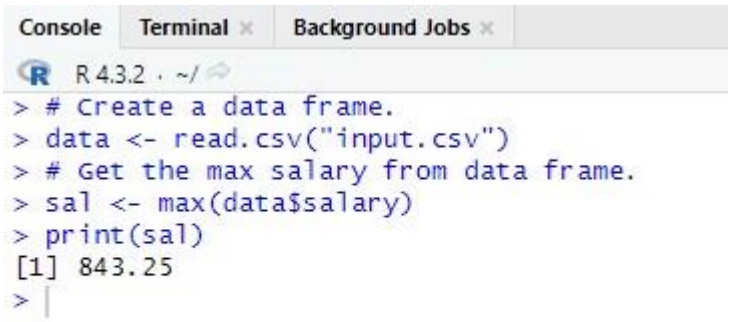
4. # Create a data frame.

   data <- read.csv("input.csv")

   # Get the max salary from data frame.

   sal <- max(data$salary)

   print(sal)



5. # Create a data frame.

   data <- read.csv("input.csv")

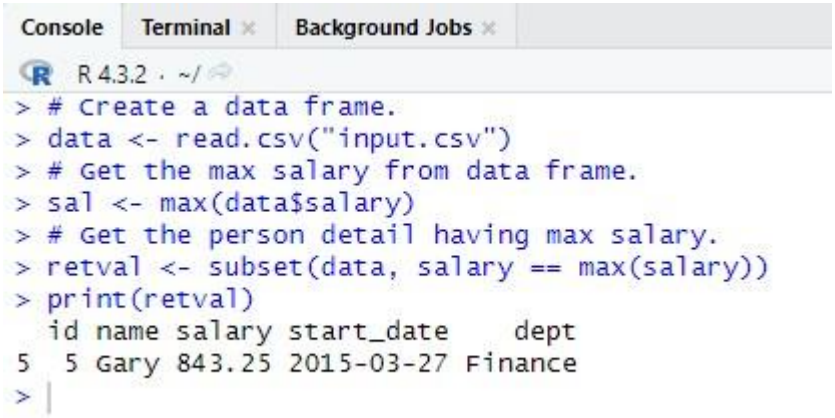
   # Get the max salary from data frame.

   sal <- max(data$salary)

   # Get the person detail having max salary.

   retval <- subset(data, salary == max(salary))

   print(retval)

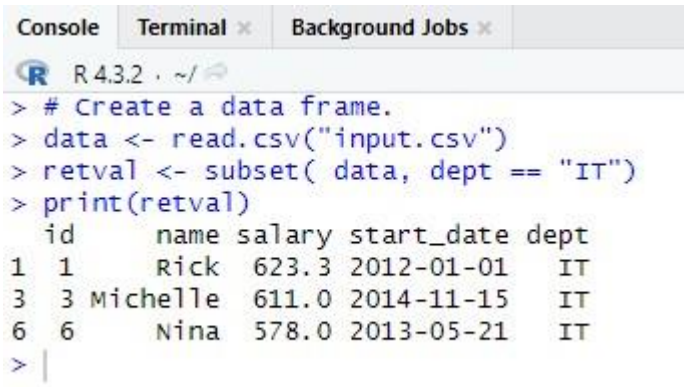


6. # Create a data frame.

   data <- read.csv("input.csv")

   retval <- subset(data, dept == "IT")

   print(retval)

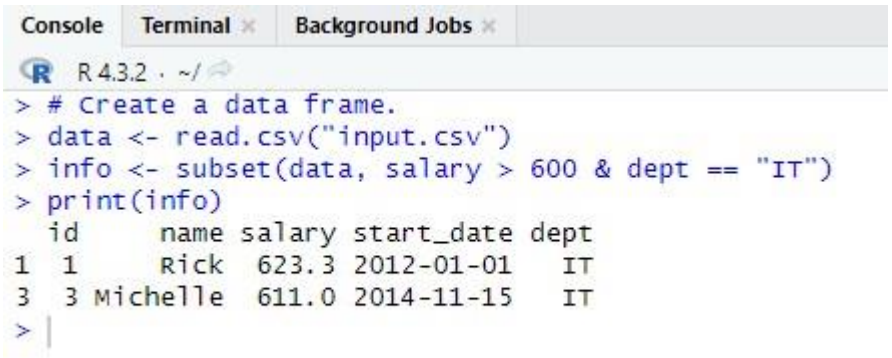


7. # Create a data frame.

   data <- read.csv("input.csv")

   info <- subset(data, salary > 600 & dept == "IT")

   print(info)

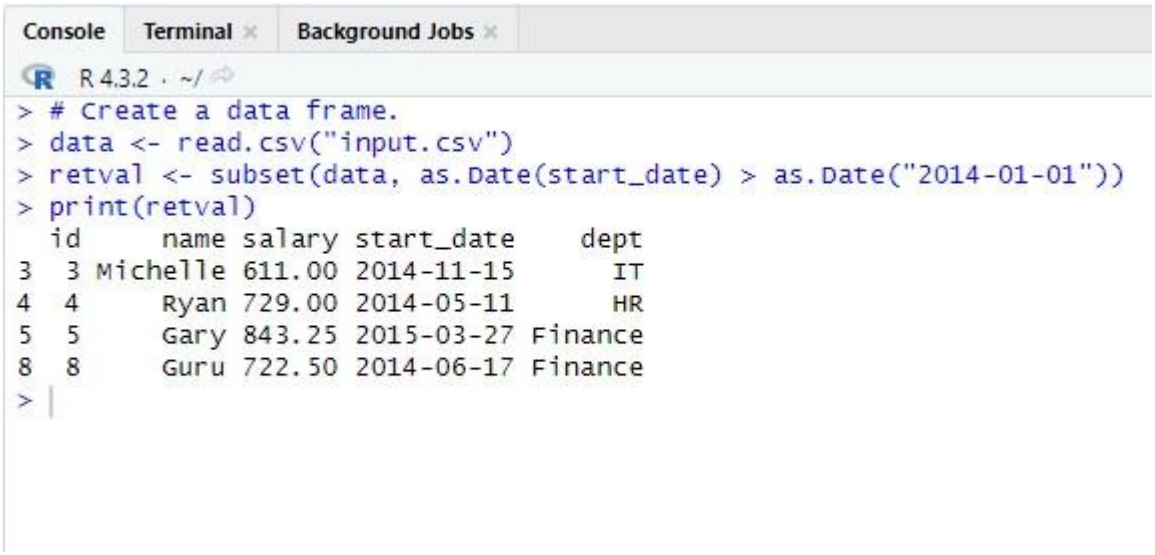


8. # Create a data frame.

   data <- read.csv("input.csv")

   retval <- subset(data, as.Date(start_date) > as.Date("2014-01-01"))

   print(retval)



9. # Create a data frame.

   data <- read.csv("input.csv")

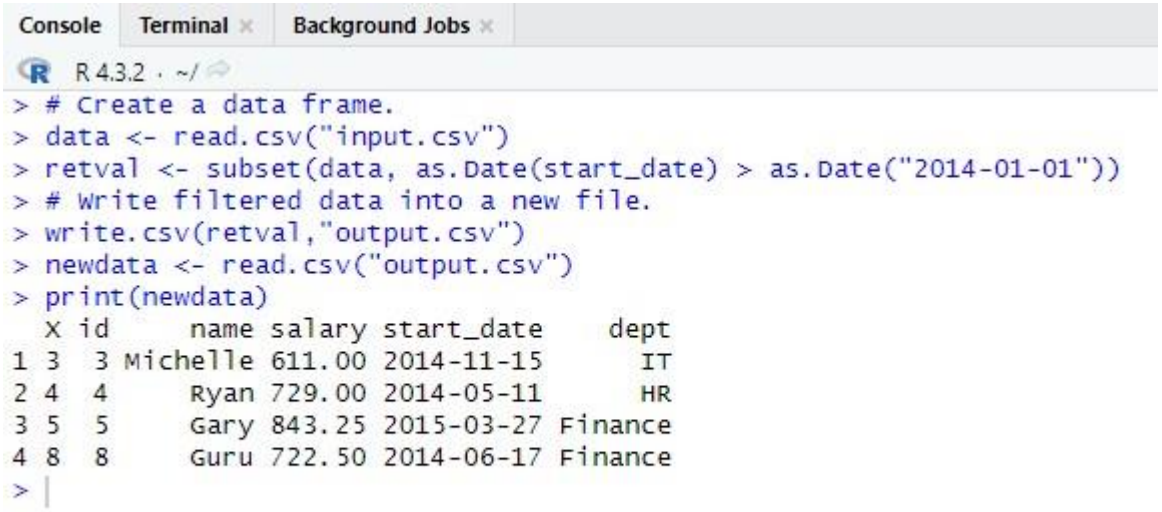
   retval <- subset(data, as.Date(start_date) > as.Date("2014-01-01"))

   # Write filtered data into a new file.

   write.csv(retval,"output.csv")

   newdata <- read.csv("output.csv")

   print(newdata)



10.# Create a data frame.

data <- read.csv("input.csv")

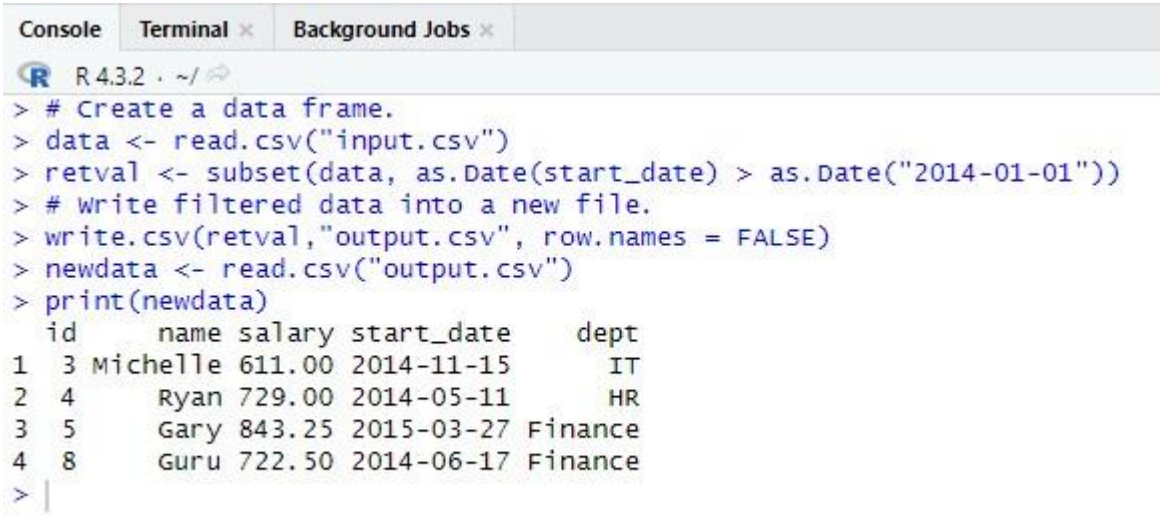
retval <- subset(data, as.Date(start_date) > as.Date("2014-01-01"))

Write filtered data into a new file.

write.csv(retval,"output.csv", row.names = FALSE)

newdata <- read.csv("output.csv")

print(newdata)



INSTITUTION: KCA University

NAME: Raphael Kang’eri

UNIT NAME: Programming for Data Science

UNIT CODE: BSD 320

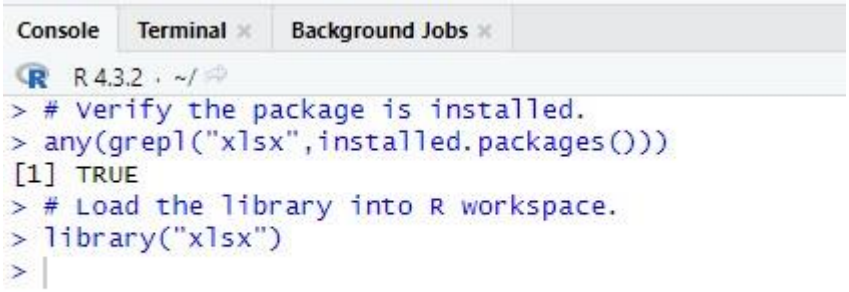
R-EXCEL

1. # Verify the package is installed.

   any(grepl("xlsx",installed.packages()))

   # Load the library into R workspace.

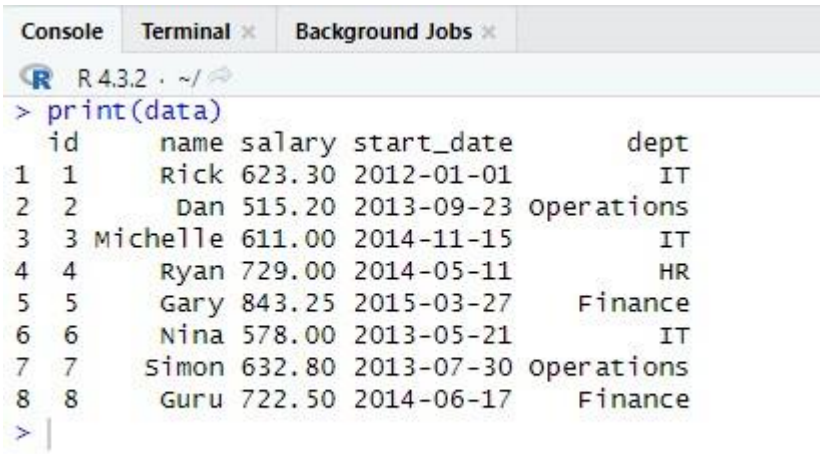
   library("xlsx")



2. # Read the first worksheet in the file input.xlsx.

   data <- read.xlsx("input.xlsx", sheetIndex = 1)

   print(data)



f-scientific-computing-with-python

February 8, 2024

1 NUMPY

[3]:
```
#heterogenous 1D array
arr = [1,2,"a",3,"b"]
```

[8]:
```
import numpy as np
x = np.array([1,2,3])
y = np.array([4,5,6])
x*y
```

[8]:
```
array([ 4, 10, 18])
```

[9]:
```
import numpy as np
#define
just_a_list = [1,2,3]
#show
just_a_list
```

[9]:
```
[1, 2, 3]
```

[10]:
```
import numpy as np
just_a_list = [1,2,3]
np.array(just_a_list)
```

[10]:
```
array([1, 2, 3])
```

[11]:
```
import numpy as np
#define
just_a_matrix = [[1,2,3],[4,5,6],[7,8,9]]
#show
just_a_matrix
```

[11]:
```
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

[12]:
```
import numpy as np
#define
just_a_matrix = [[1,2,3],[4,5,6],[7,8,9]]
```

```
#show
np.array(just_a_matrix)
```

[12]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

[13]:
```
import numpy as np
np.arange(0,20)
```

[13]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19])
```

[14]:
```
import numpy as np
#step size of 2
np.arange(0,10,2)
```

[14]:
```
array([0, 2, 4, 6, 8])
```

[15]:
```
import numpy as np
np.zeros(4)
```

[15]:
```
array([0., 0., 0., 0.])
```

[16]:
```
import numpy as np
np.zeros((3,3))
```

[16]:
```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

[17]:
```
import numpy as np
np.ones(4)
```

[17]:
```
array([1., 1., 1., 1.])
```

[18]:
```
import numpy as np
np.ones((3,3))
```

[18]:
```
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])
```

[20]:
```
import numpy as np
#generates three numbers
#between range 0 to 10
np.linspace(0,10,3)
```

[20]:
```
array([ 0.,  5., 10.])
```

[21]:
```
import numpy as np
np.linspace(0,10,20)
```

[21]:
```
array([ 0.        ,  0.52631579,  1.05263158,  1.57894737,  2.10526316,
        2.63157895,  3.15789474,  3.68421053,  4.21052632,  4.73684211,
        5.26315789,  5.78947368,  6.31578947,  6.84210526,  7.36842105,
        7.89473684,  8.42105263,  8.94736842,  9.47368421, 10.        ])
```

[22]:
```
import numpy as np
np.eye(3)
```

[22]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

[23]:
```
import numpy as np
np.random.rand(2)
```

[23]:
```
array([0.75261764, 0.40355455])
```

[24]:
```
import numpy as np
np.random.rand(3,3)
```

[24]:
```
array([[0.47391322, 0.96980881, 0.03119947],
       [0.70224136, 0.46984861, 0.13303997],
       [0.52404209, 0.0291802 , 0.57177518]])
```

[25]:
```
import numpy as np
np.random.randn(2)
```

[25]:
```
array([0.03599606, 1.42227488])
```

[26]:
```
import numpy as np
np.random.randn(4,4)
```

[26]:
```
array([[-0.23232824,  1.52396446,  1.27804697,  1.54168661],
       [ 0.18651272, -1.01737444,  0.60560337,  0.18635974],
       [-0.34886112, -1.64104164, -0.33407226,  0.12429816],
       [ 0.09587071,  0.82793611, -0.13187869, -1.70646665]])
```

[27]:
```
import numpy as np
np.random.randint(1,100)
```

[27]:
```
57
```

[28]:
```
import numpy as np
#generates 10 random
#integers between 1 to 100
np.random.randint(1,100,10)
```

[28]:
```
array([90,  5,  2, 38, 45, 79, 57, 73,  1, 84])
```

[29]:
```
import numpy as np
#define
arr = np.arange(16)
#show
arr
```

[29]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15])
```

[30]:
```
import numpy as np
#define
arr = np.arange(16)
#show
arr
arr.reshape(4,4)
```

[30]:
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11],
       [12, 13, 14, 15]])
```

[31]:
```
import numpy as np
#define
random_arr = np.random.randint(0,50,10)
#show
random_arr
```

[31]:
```
array([19, 10, 11,  7, 21, 35, 26,  1, 10, 24])
```

[32]:
```
import numpy as np
#define
random_arr = np.random.randint(0,50,10)
#show
random_arr
random_arr.max()
```

[32]:
```
45
```

[33]:
```
import numpy as np
#define
random_arr = np.random.randint(0,50,10)
```

```
#show
random_arr
random_arr.max()
random_arr.argmax()
```

[33]: 2

[34]:
```
import numpy as np
#define
random_arr = np.random.randint(0,50,10)
#show
random_arr
random_arr.max()
random_arr.argmax()
random_arr.min()
```

[34]: 3

[35]:
```
import numpy as np
#define
random_arr = np.random.randint(0,50,10)
#show
random_arr
random_arr.max()
random_arr.argmax()
random_arr.min()
random_arr.argmin()
```

[35]: 2

[36]:
```
import numpy as np
#define
random_arr = np.random.randint(0,50,10)
#show
random_arr
arr.shape
```

[36]: (16,)

[37]:
```
arr.reshape(1,16)
```

[37]:
```
array([[ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15]])
```

[38]:
```
arr.reshape(1,16).shape
```

[38]: (1, 16)

[39]:
```
arr.reshape(16,1)
```

[39]:
```
array([[ 0],
       [ 1],
       [ 2],
       [ 3],
       [ 4],
       [ 5],
       [ 6],
       [ 7],
       [ 8],
       [ 9],
       [10],
       [11],
       [12],
       [13],
       [14],
       [15]])
```

[40]:
```
arr.reshape(16,1).shape
```

[40]:
```
(16, 1)
```

[41]:
```
arr.dtype
```

[41]:
```
dtype('int32')
```

[42]:
```
#Get a value
#at an index
arr[5]
```

[42]:
```
5
```

[43]:
```
#Get values
#in a range
arr[1:5]
```

[43]:
```
array([1, 2, 3, 4])
```

[44]:
```
#Setting a value with
#index range
arr[0:5]=100#Show
arr
```

[44]:
```
array([100, 100, 100, 100, 100,   5,   6,   7,   8,   9,  10,  11,  12,
        13,  14,  15])
```

[45]:
```
#Reset array, we'll
#see the reason behind it soon
arr = np.arange(0,11)#Show
arr
```

[45]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

[46]:
```
slice_of_arr = arr[0:6]#Show slice
slice_of_arr
```

[46]:
```
array([0, 1, 2, 3, 4, 5])
```

[47]:
```
#Change Slice
slice_of_arr[:]=99#Show Slice again
slice_of_arr
```

[47]:
```
array([99, 99, 99, 99, 99, 99])
```

[48]:
```
arr
```

[48]:
```
array([99, 99, 99, 99, 99, 99,  6,  7,  8,  9, 10])
```

[51]:
```
a = [[1,2,3],[4,5,6]]#Converting to
#numpy 2d array
np_a = np.array(a)#show
np_a
```

[51]:
```
array([[1, 2, 3],
       [4, 5, 6]])
```

[52]:
```
#2 rows, 3 columns
np_a.shape
```

[52]:
```
(2, 3)
```

[53]:
```
b = [100,200,300]#converting to
#numpy array
np_b = np.array(b)#show
np_b
```

[53]:
```
array([100, 200, 300])
```

[54]:
```
np_b.shape
```

[54]:
```
(3,)
```

[55]:
```
np_b_reshaped = np_b.reshape(1,3)#show
np_b_reshaped
```

[55]:
```
array([[100, 200, 300]])
```

[56]:
```
#1 row, 3 columns
np_b_reshaped.shape
```

[56]:
```
(1, 3)
```

[57]:
```
np_a + np_b_reshaped
```

[57]:
```
array([[101, 202, 303],
       [104, 205, 306]])
```

[58]:
```
np.array([[100,200,300],[100,200,300]])
```

[58]:
```
array([[100, 200, 300],
       [100, 200, 300]])
```

[59]:
```
arr_2d = np.array(([1,2,3],[4,5,6],[7,8,9]))#Show
arr_2d
```

[59]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

[60]:
```
#Indexing row
arr_2d[1]
```

[60]:
```
array([4, 5, 6])
```

[61]:
```
#Getting individual element value
arr_2d[1][0]
```

[61]:
```
4
```

[62]:
```
#another way
arr_2d[1,0]
```

[62]:
```
4
```

[71]:
```
arr_2d[:2, 1:]  # Correct slicing for the top right corner
```

[71]:
```
array([[2, 3],
       [5, 6]])
```

[70]:
```
#Shape of the
#bottom row
arr_2d[2]
```

[70]:
```
array([7, 8, 9])
```

[72]:
```
arr1 = np.arange(1,11)#show
arr1
```

[72]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

[73]:
```
arr1 > 5
```

[73]:
```
array([False, False, False, False, False,  True,  True,  True,  True,
        True])
```

[74]:
```
bool_arr = arr1 > 5#show
bool_arr
```

[74]:
```
array([False, False, False, False, False,  True,  True,  True,  True,
        True])
```

[75]:
```
arr1[bool_arr]
```

[75]:
```
array([ 6,  7,  8,  9, 10])
```

[76]:
```
arr1[arr1 > 2]
```

[76]:
```
array([ 3,  4,  5,  6,  7,  8,  9, 10])
```

[78]:
```
arr2 = np.arange(0,10)
arr2 + arr2
```

[78]:
```
array([ 0,  2,  4,  6,  8, 10, 12, 14, 16, 18])
```

[79]:
```
arr2 * arr2
```

[79]:
```
array([ 0,  1,  4,  9, 16, 25, 36, 49, 64, 81])
```

[80]:
```
arr2 - arr2
```

[80]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

[81]:
```
arr2 / arr2
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_9100\3151952473.py:1: RuntimeWarning:
invalid value encountered in divide
  arr2 / arr2
```

[81]:
```
array([nan,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.])
```

[82]:
```
1 / arr2
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_9100\1481894037.py:1: RuntimeWarning:
divide by zero encountered in divide
  1 / arr2
```

[82]:
```
array([       inf, 1.        , 0.5       , 0.33333333, 0.25      ,
       0.2       , 0.16666667, 0.14285714, 0.125     , 0.11111111])
```

[83]:
```
arr2 ** 3
```

[83]:
```
array([  0,   1,   8,  27,  64, 125, 216, 343, 512, 729], dtype=int32)
```

[84]:
```
np.sqrt(arr2)
```

[84]:
```
array([0.        , 1.        , 1.41421356, 1.73205081, 2.        ,
       2.23606798, 2.44948974, 2.64575131, 2.82842712, 3.        ])
```

[85]:
```
np.exp(arr2)
```

[85]:
```
array([1.00000000e+00, 2.71828183e+00, 7.38905610e+00, 2.00855369e+01,
       5.45981500e+01, 1.48413159e+02, 4.03428793e+02, 1.09663316e+03,
       2.98095799e+03, 8.10308393e+03])
```

[86]:
```
np.max(arr2)
```

[86]:
```
9
```

[87]:
```
np.sin(arr2)
```

[87]:
```
array([ 0.        ,  0.84147098,  0.90929743,  0.14112001, -0.7568025 ,
       -0.95892427, -0.2794155 ,  0.6569866 ,  0.98935825,  0.41211849])
```

[88]:
```
np.log(arr2)
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_9100\3827996858.py:1: RuntimeWarning:
divide by zero encountered in log
  np.log(arr2)
```

[88]:
```
array([      -inf, 0.        , 0.69314718, 1.09861229, 1.38629436,
       1.60943791, 1.79175947, 1.94591015, 2.07944154, 2.19722458])
```

[90]:
```
import time
import numpy as np

a = np.random.rand(1000000)
```

```
b = np.random.rand(1000000)

tic = time.time()
c = np.dot(a, b)  # dot product of two vectors
toc = time.time()

print("Time taken in vectorized version: " + str(1000 * (toc - tic)) + " ms") ␣
 ↪# Corrected string formatting
```

```
Time taken in vectorized version: 199.8136043548584 ms
```

[92]:
```
c = 0
tic = time.time()
for i in range(1000000):
    c += a[i] * b[i]
toc = time.time()
print("Time taken in for loop version: " + str(1000 * (toc - tic)) + " ms")  #␣
 ↪Corrected string formatting
```

```
Time taken in for loop version: 690.1640892028809 ms
```

[ ]:
```

```

matplotlib

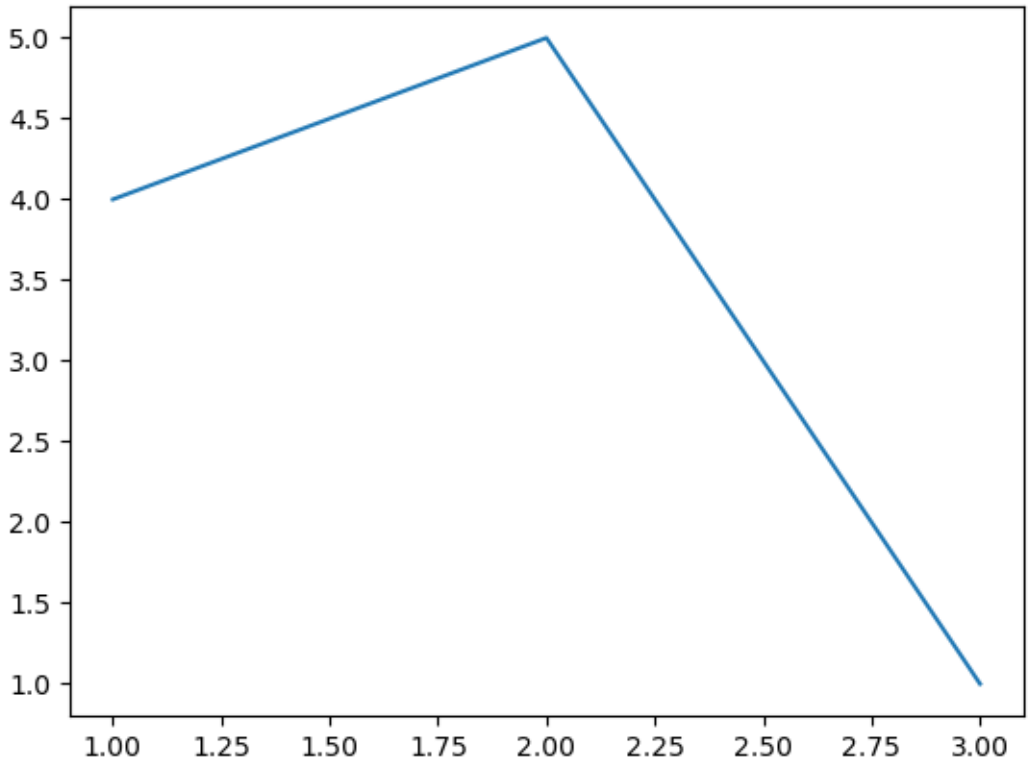
February 8, 2024

1 MATPLOTLIB

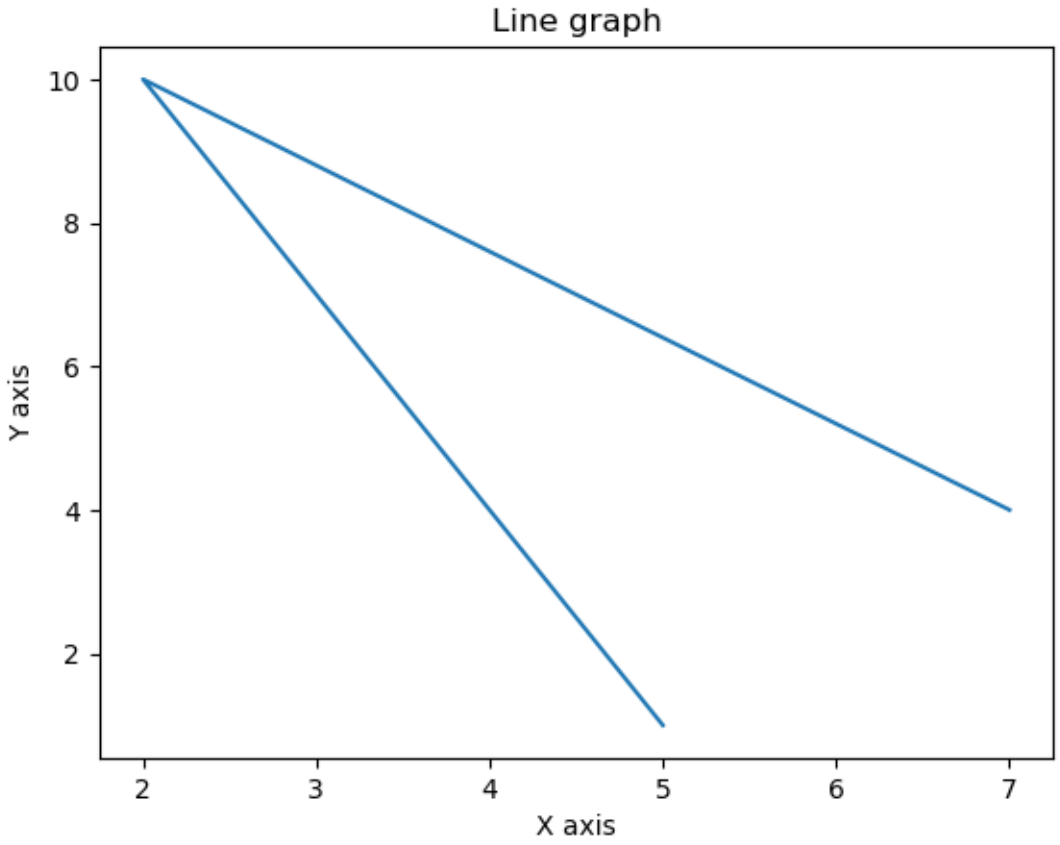
[1]:
```
import matplotlib
matplotlib.__version__
```

[1]:
```
'3.7.2'
```

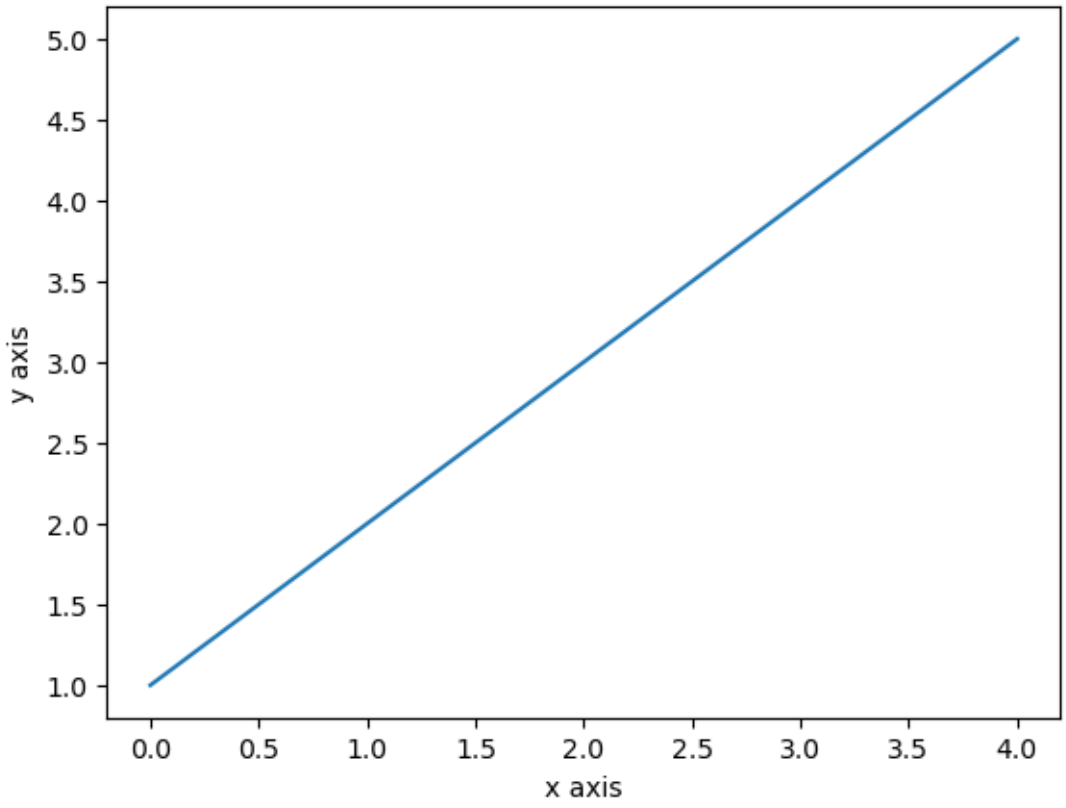
[2]:
```
from matplotlib import pyplot as plt
#ploting our canvas
plt.plot([1,2,3],[4,5,1])
#display the graph
plt.show()
```



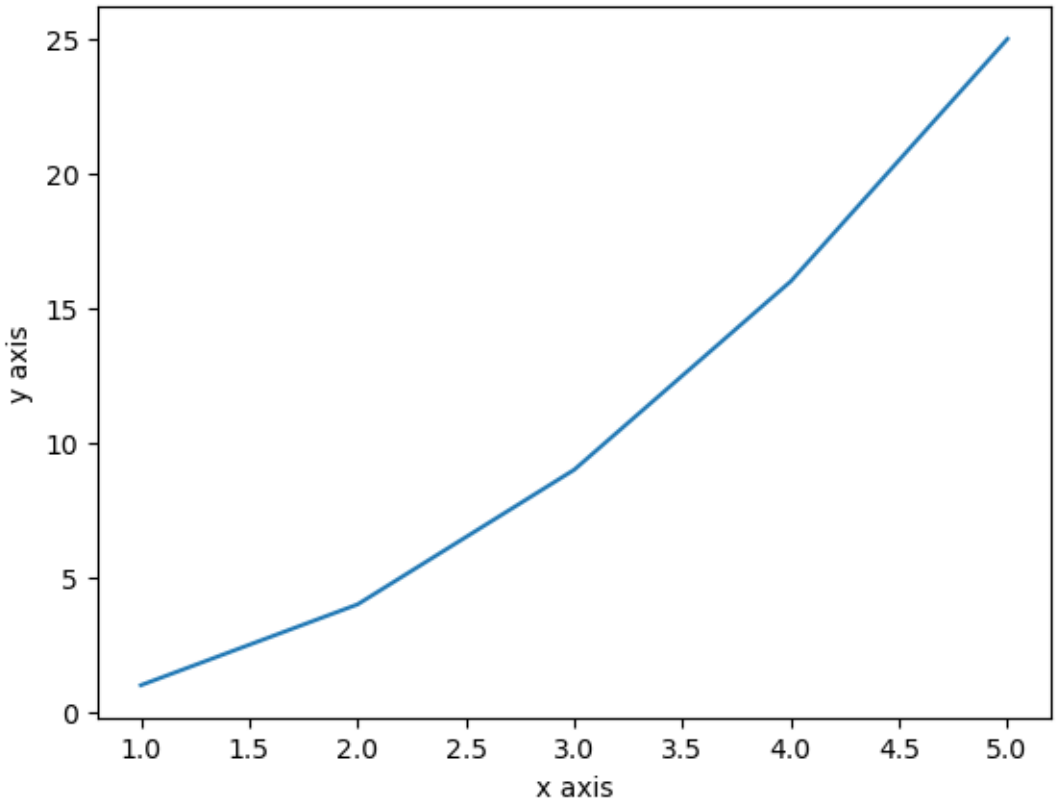
[3]:
```
x = [5, 2, 7]
y = [1, 10, 4]
plt.plot(x, y)
plt.title('Line graph')
plt.ylabel('Y axis')
plt.xlabel('X axis')
plt.show()
```



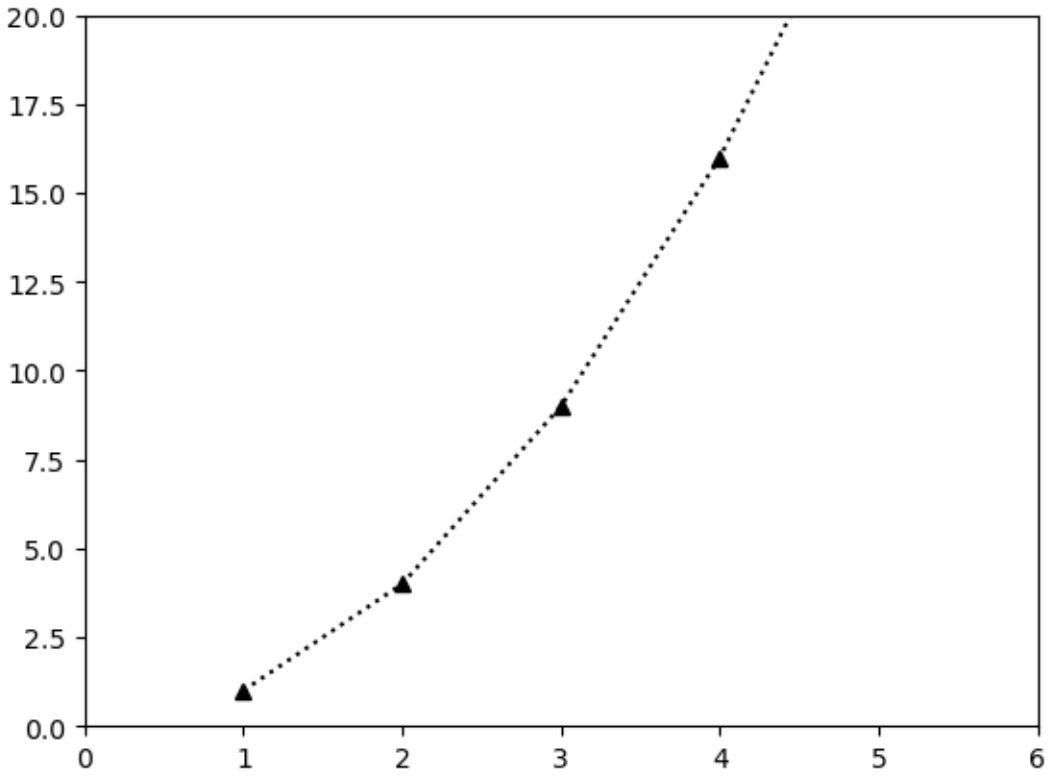
[4]:
```
from matplotlib import pyplot as plt
plt.plot([1,2,3,4,5])
plt.ylabel("y axis")
plt.xlabel('x axis')
plt.show()
```



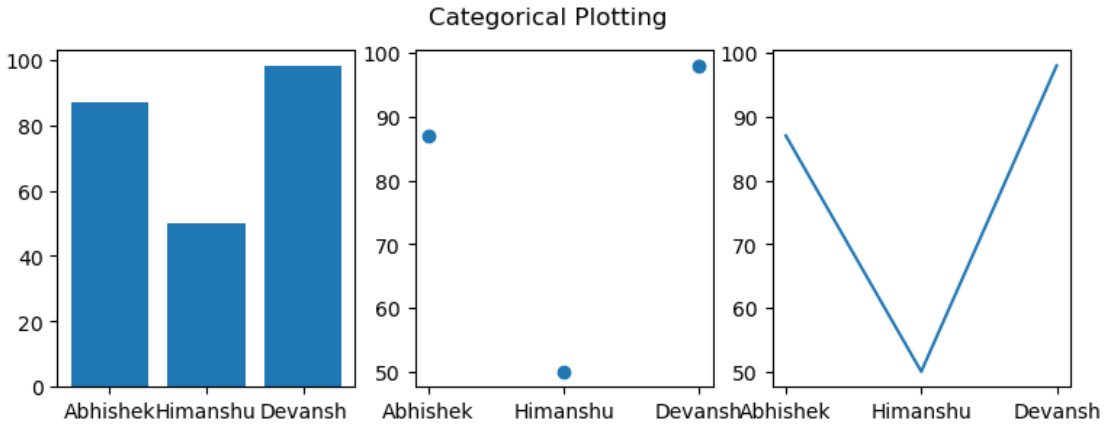
[5]:
```
from matplotlib import pyplot as plt
plt.plot([1,2,3,4,5],[1,4,9,16,25])
plt.ylabel("y axis")
plt.xlabel('x axis')
plt.show()
```



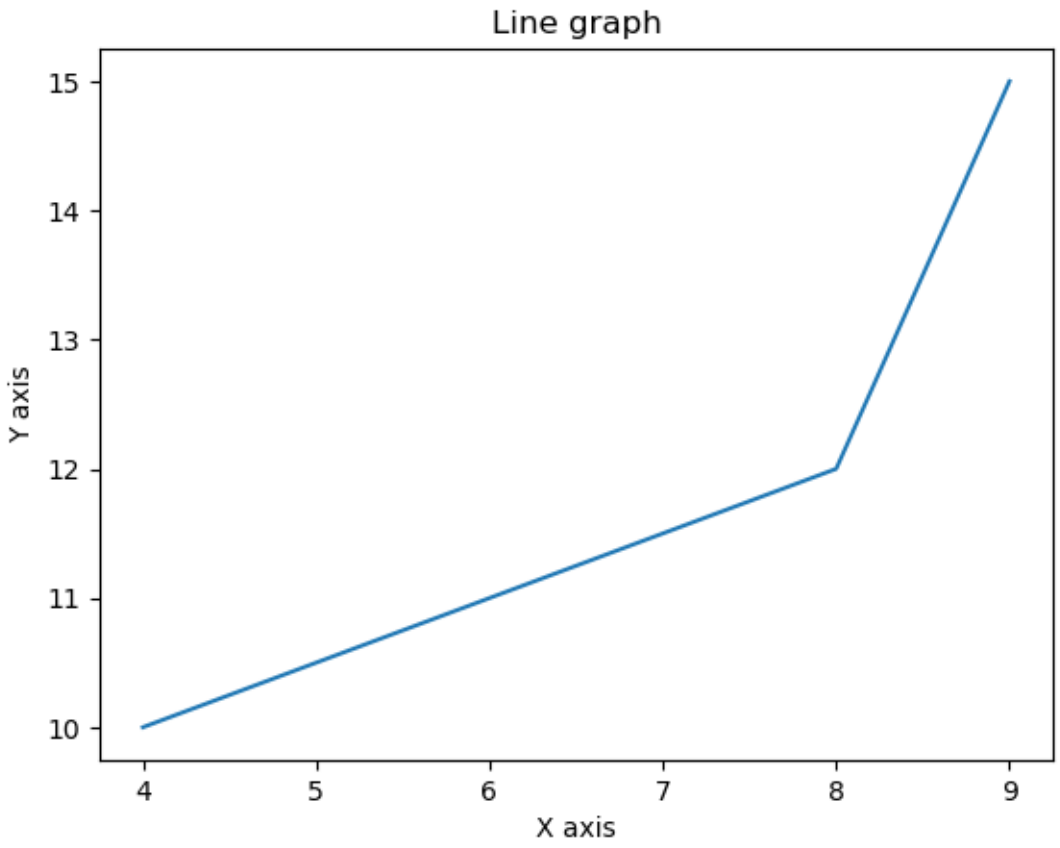
[11]:
```
from matplotlib import pyplot as plt
plt.plot([1, 2, 3, 4,5], [1, 4, 9, 16,25], '^k:')
plt.axis([0, 6, 0, 20])
plt.show()
```



[12]:
```
from matplotlib import pyplot
names = ['Abhishek', 'Himanshu', 'Devansh']
marks= [87,50,98]

plt.figure(figsize=(9,3))

plt.subplot(131)
plt.bar(names, marks)
plt.subplot(132)
plt.scatter(names, marks)
plt.subplot(133)
plt.plot(names, marks)
plt.suptitle('Categorical Plotting')
plt.show()
```

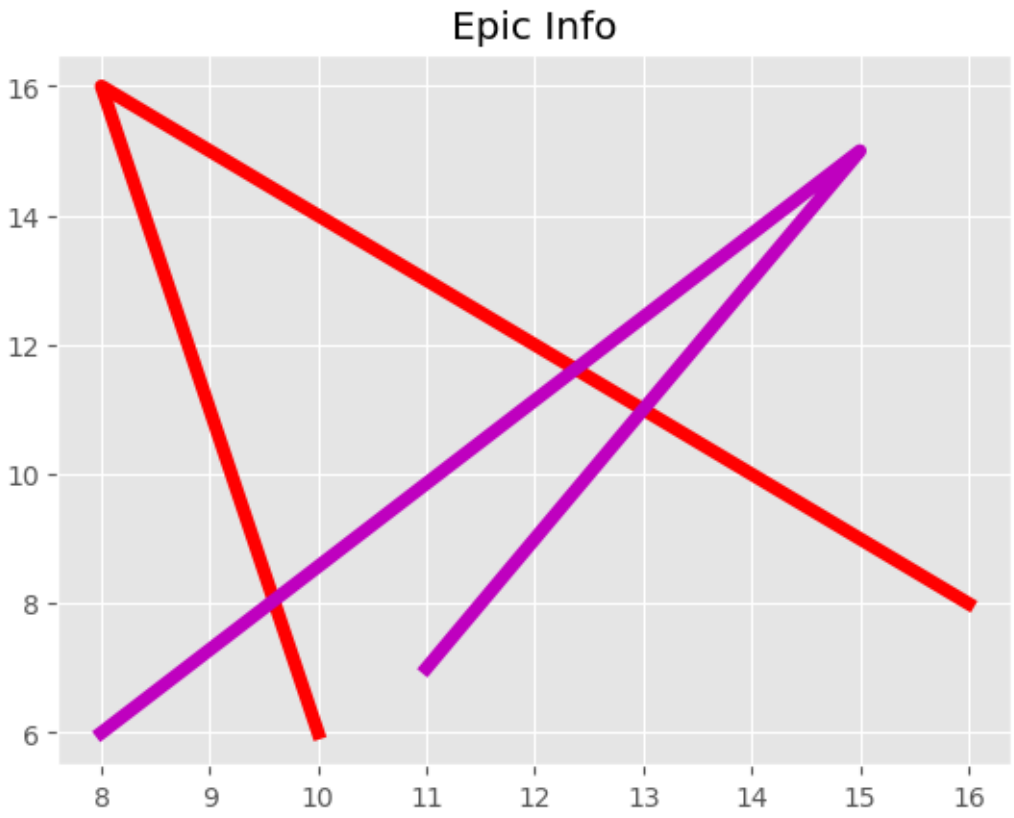


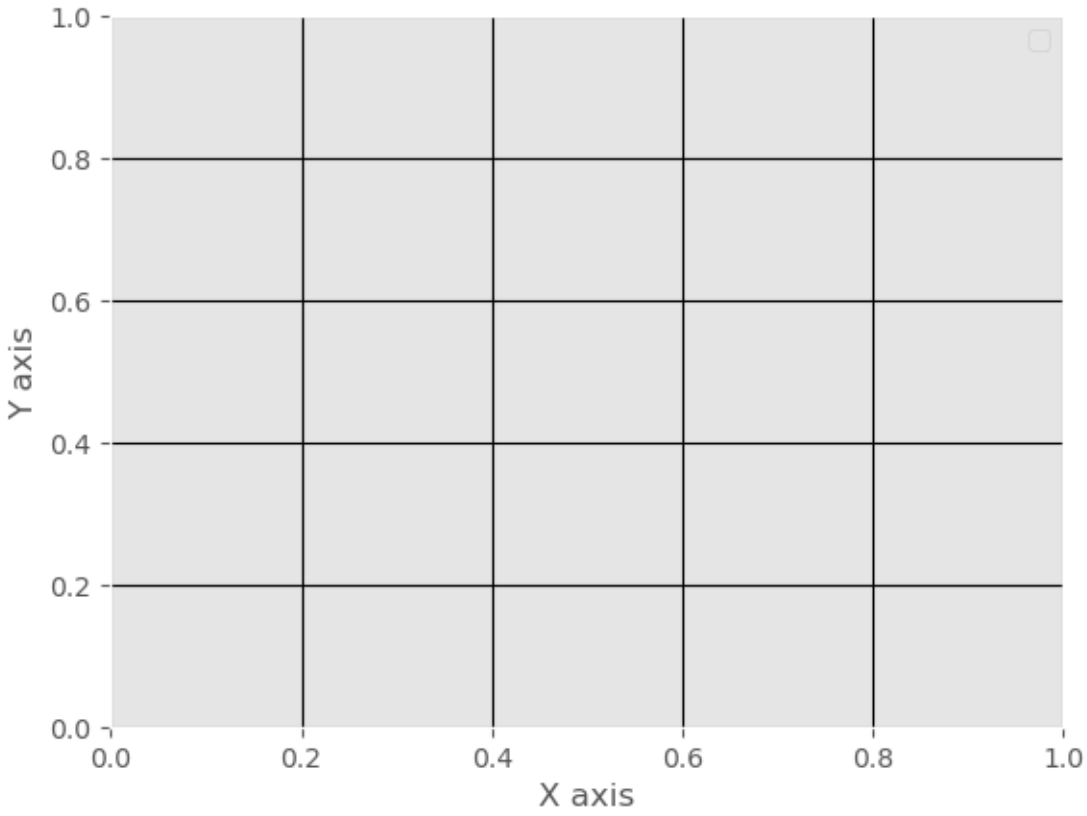
[13]:
```
from matplotlib import pyplot as plt

x = [4,8,9]
y = [10,12,15]

plt.plot(x,y)

plt.title("Line graph")
plt.ylabel('Y axis')
plt.xlabel('X axis')
plt.show()
```



[14]:
```
from matplotlib import pyplot as plt
from matplotlib import style

style.use('ggplot')
x = [16, 8, 10]
y = [8, 16, 6]
x2 = [8, 15, 11]
y2 = [6, 15, 7]
plt.plot(x, y, 'r', label='line one', linewidth=5)
plt.plot(x2, y2, 'm', label='line two', linewidth=5)
plt.title('Epic Info')
fig = plt.figure()
plt.ylabel('Y axis')
plt.xlabel('X axis')
plt.legend()
plt.grid(True, color='k')
plt.show()
```

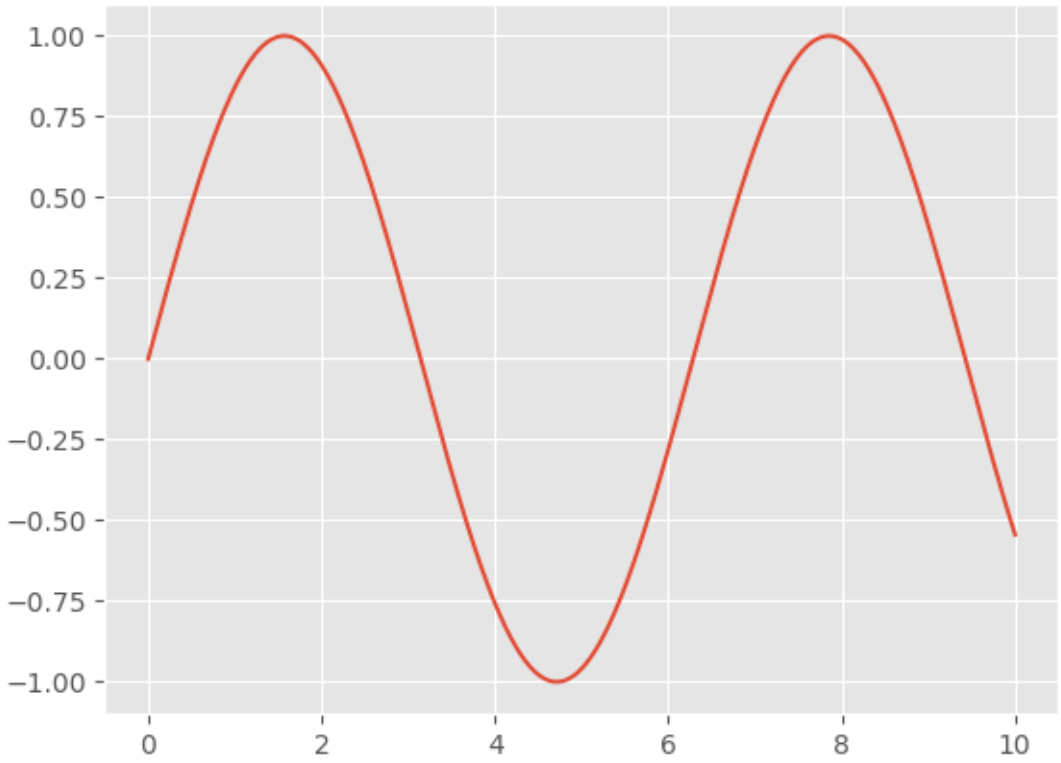
```
No artists with labels found to put in legend.  Note that artists whose label
start with an underscore are ignored when legend() is called with no argument.
```


Epic Info
16
14
12
10
8
6
8
9
10
11
12
13
14
15
16



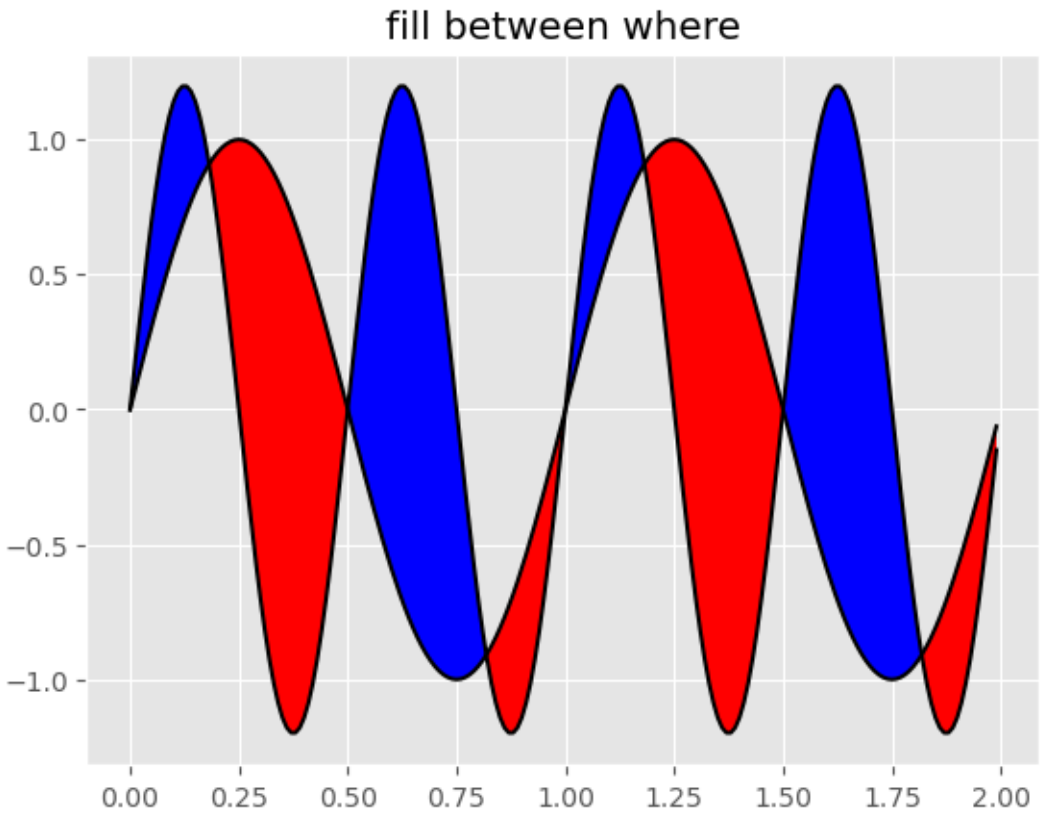
[15]:
```
import numpy as np
import matplotlib.pyplot as plt

fig = plt.figure()
ax = plt.axes()

x = np.linspace(0, 10, 1000)
ax.plot(x, np.sin(x))
```

[15]:
```
[<matplotlib.lines.Line2D at 0x18f3403a790>]
```

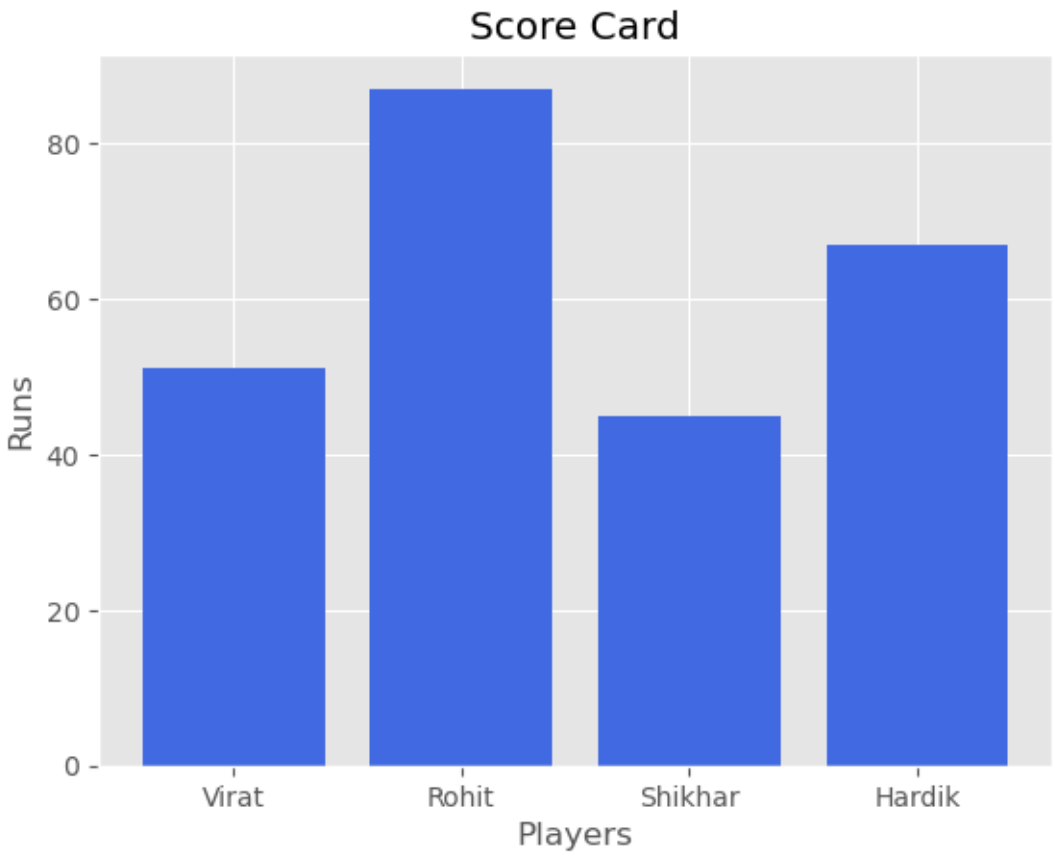


[17]:
```
import matplotlib.pyplot as plt
import numpy as np

x = np.arange(0.0, 2, 0.01)
y1 = np.sin(2 * np.pi * x)
y2 = 1.2 * np.sin(4 * np.pi * x)
fig, ax = plt.subplots(1, sharex=True)
ax.plot(x, y1, x, y2, color='black')
ax.fill_between(x, y1, y2, where=y2 >= y1, facecolor='blue', interpolate=True)
ax.fill_between(x, y1, y2, where=y2 <= y1, facecolor='red', interpolate=True)
ax.set_title('fill between where')
```

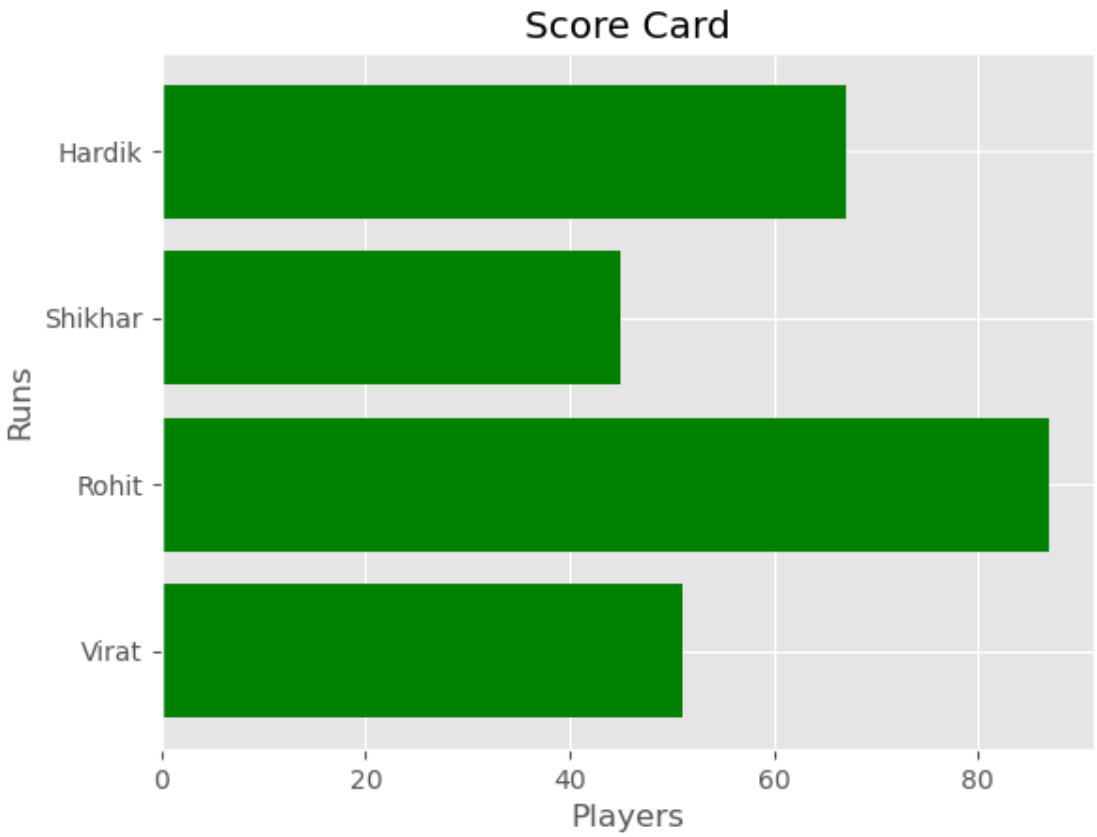
[17]:
```
Text(0.5, 1.0, 'fill between where')
```



[20]:
```
from matplotlib import pyplot as plt
players = ['Virat','Rohit','Shikhar','Hardik']
runs = [51,87,45,67]
plt.bar(players,runs,color = '#4169E1')
plt.title('Score Card')
plt.xlabel('Players')
plt.ylabel('Runs')
plt.show()
```

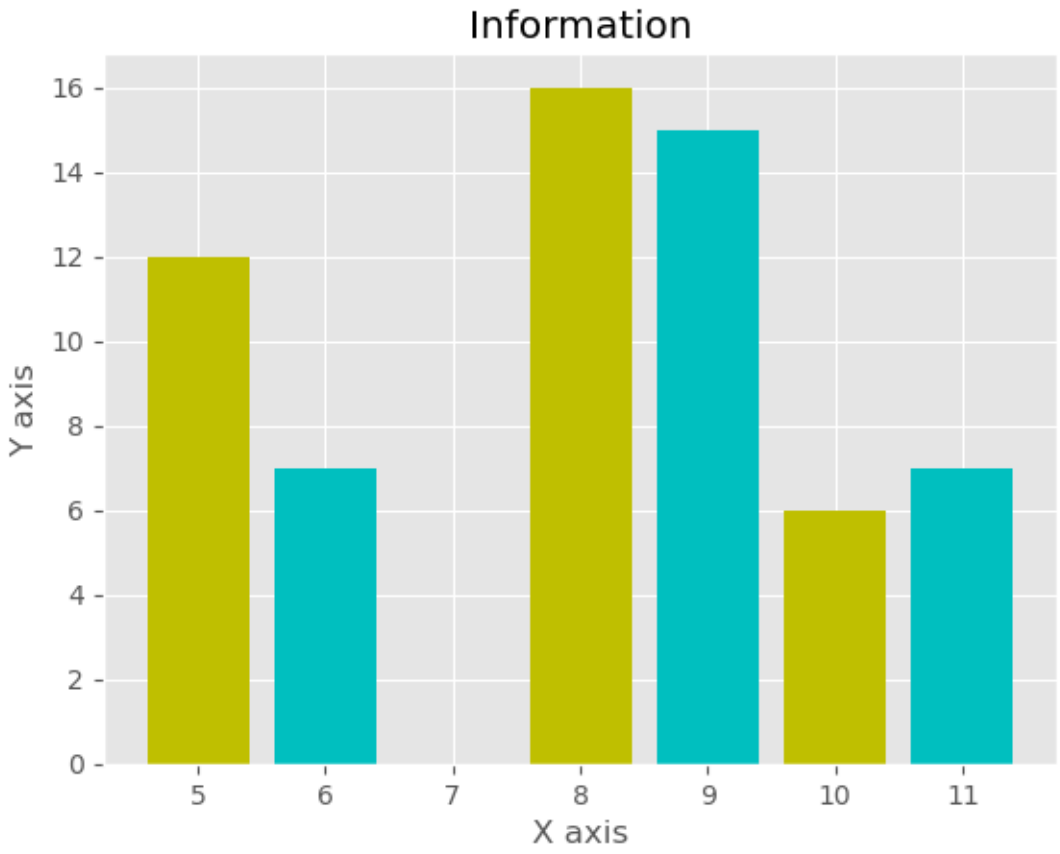


[21]:
```
from matplotlib import pyplot as plt
players = ['Virat','Rohit','Shikhar','Hardik']
runs = [51,87,45,67]
plt.barh(players,runs, color = 'green')
plt.title('Score Card')
plt.xlabel('Players')
plt.ylabel('Runs')
plt.show()
```



[22]:
```
from matplotlib import pyplot as plt
from matplotlib import style

style.use('ggplot')

x = [5,8,10]
y = [12,16,6]

x2 = [6,9,11]
y2 = [7,15,7]


plt.bar(x, y, color = 'y', align='center')
plt.bar(x2, y2, color='c', align='center')

plt.title('Information')

plt.ylabel('Y axis')
plt.xlabel('X axis')
```

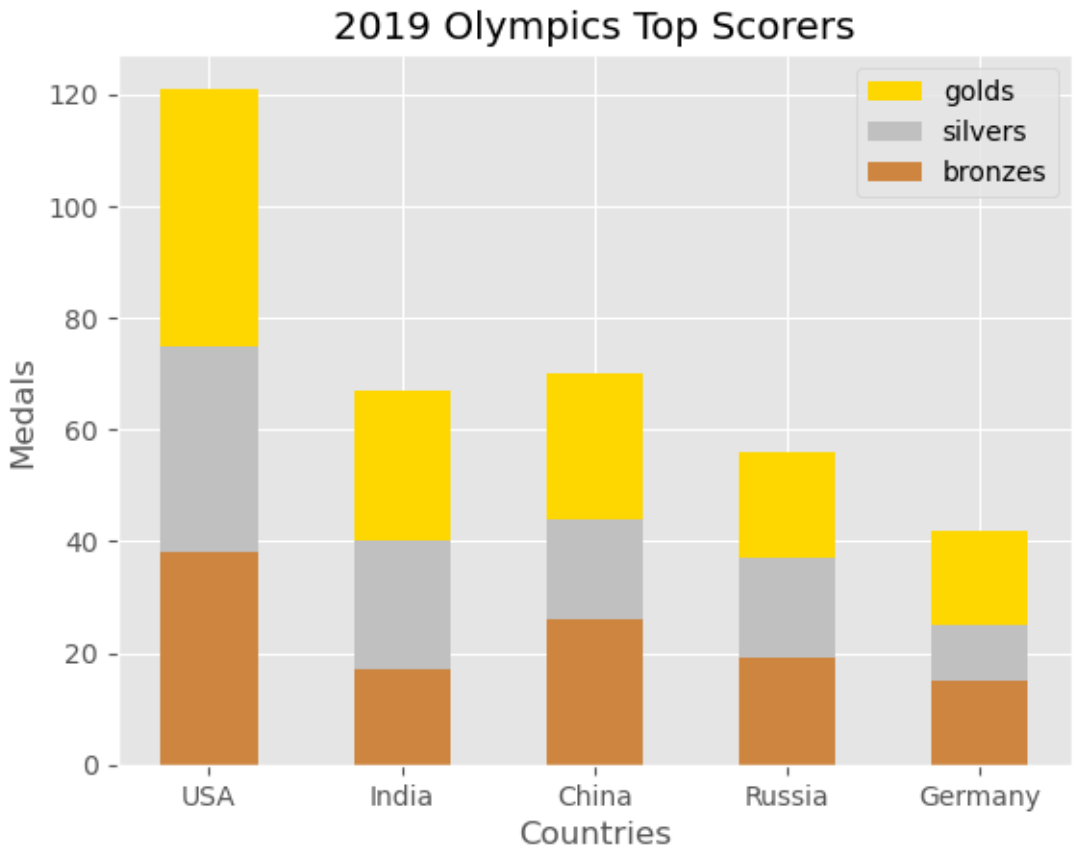
[22]: Text(0.5, 0, 'X axis')



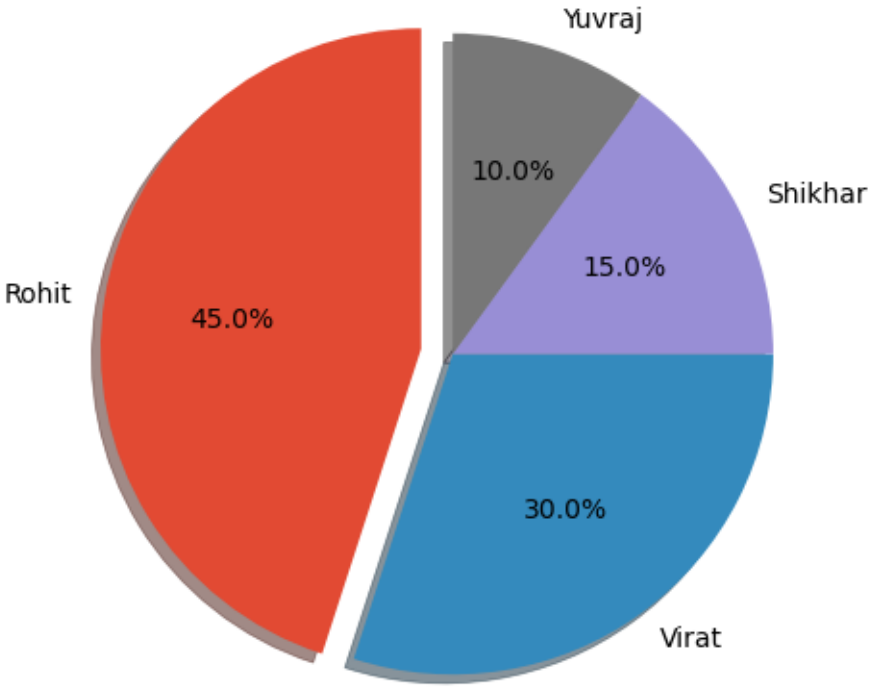
[23]:
```
from matplotlib import pyplot as plt
import numpy as np

countries = ['USA', 'India', 'China', 'Russia', 'Germany']
bronzes = np.array([38, 17, 26, 19, 15])
silvers = np.array([37, 23, 18, 18, 10])
golds = np.array([46, 27, 26, 19, 17])
ind = [x for x, _ in enumerate(countries)]

plt.bar(ind, golds, width=0.5, label='golds', color='gold',␣
 ↪bottom=silvers+bronzes)
plt.bar(ind, silvers, width=0.5, label='silvers', color='silver',␣
 ↪bottom=bronzes)
plt.bar(ind, bronzes, width=0.5, label='bronzes', color='#CD853F')

plt.xticks(ind, countries)
plt.ylabel("Medals")
plt.xlabel("Countries")
```

```
plt.legend(loc="upper right")
plt.title("2019 Olympics Top Scorers")
```

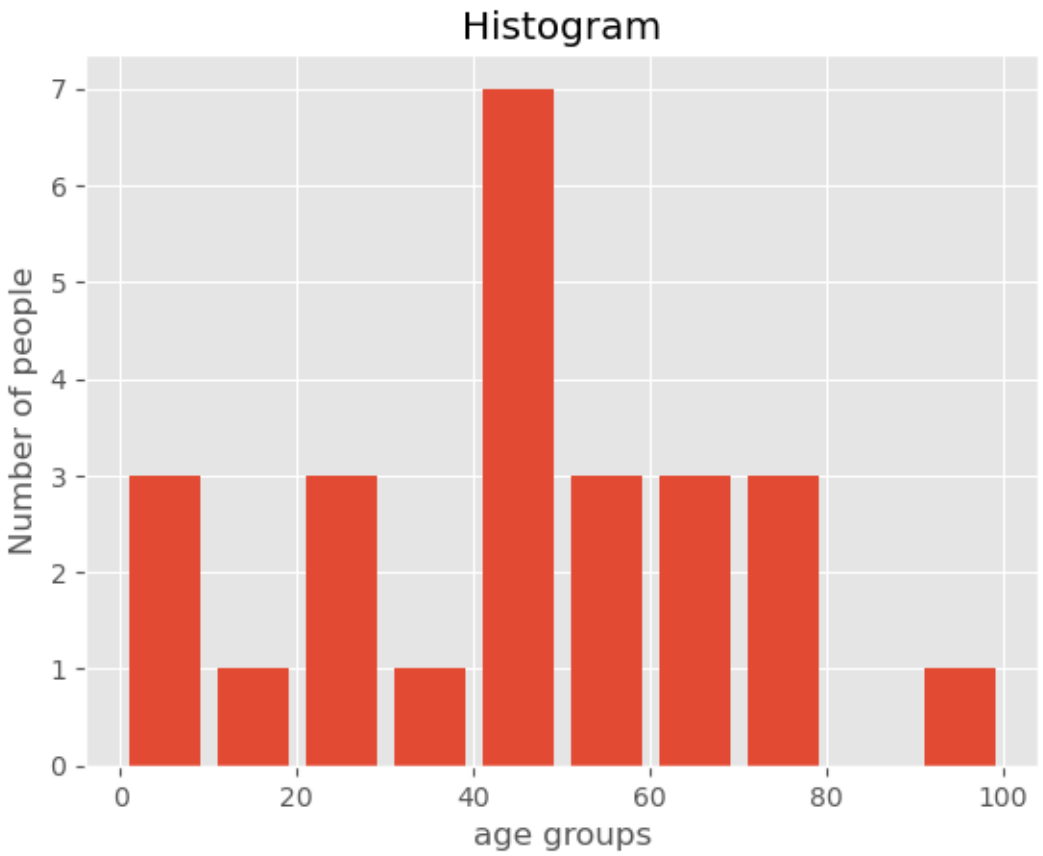
[23]: Text(0.5, 1.0, '2019 Olympics Top Scorers')



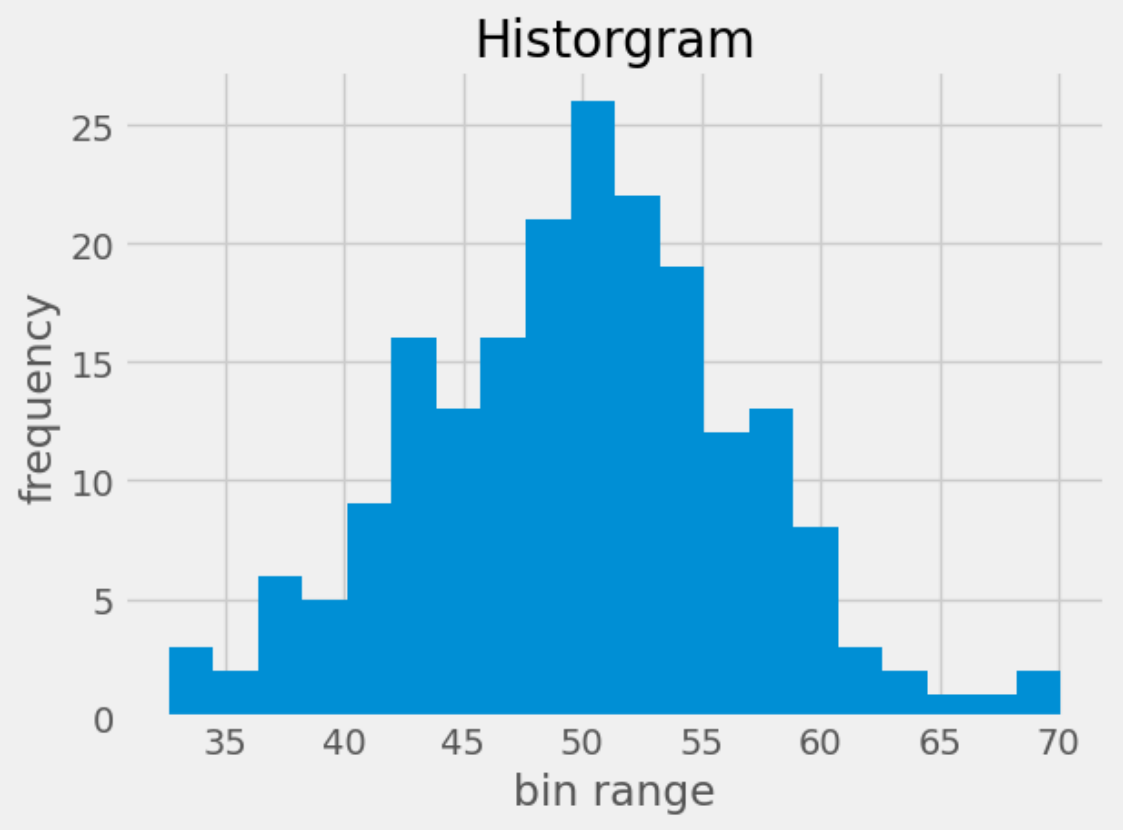
[24]:
```
from matplotlib import pyplot as plt

# Pie chart, where the slices will be ordered and plotted counter-clockwise:
Players = 'Rohit', 'Virat', 'Shikhar', 'Yuvraj'
Runs = [45, 30, 15, 10]
explode = (0.1, 0, 0, 0) # it "explode" the 1st slice

fig1, ax1 = plt.subplots()
ax1.pie(Runs, explode=explode, labels=Players, autopct='%1.1f%%',
 shadow=True, startangle=90)
ax1.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.

plt.show()
```



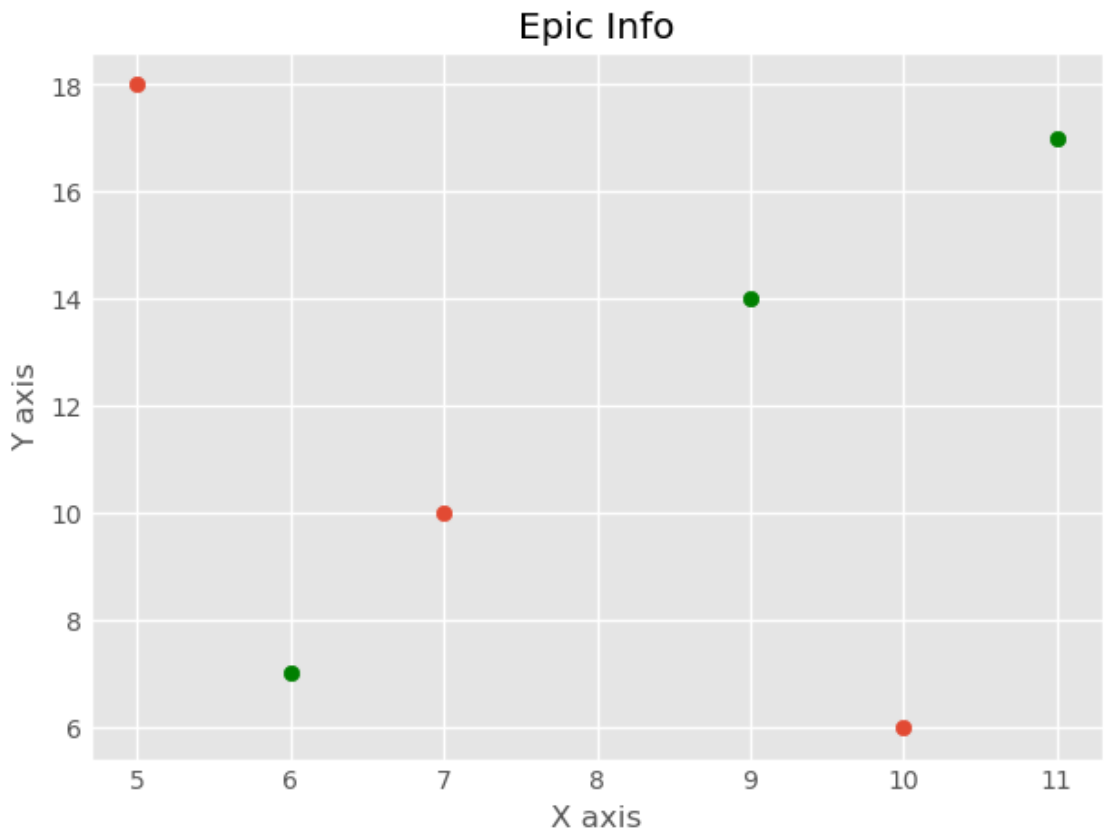
[25]:
```
from matplotlib import pyplot as plt
from matplotlib import pyplot as plt
population_age =␣
 ↪[21,53,60,49,25,27,30,42,40,1,2,102,95,8,15,105,70,65,55,70,75,60,52,44,
43,42,45]
bins = [0,10,20,30,40,50,60,70,80,90,100]
plt.hist(population_age, bins, histtype='bar', rwidth=0.8)
plt.xlabel('age groups')
plt.ylabel('Number of people')
plt.title('Histogram')
plt.show()
```



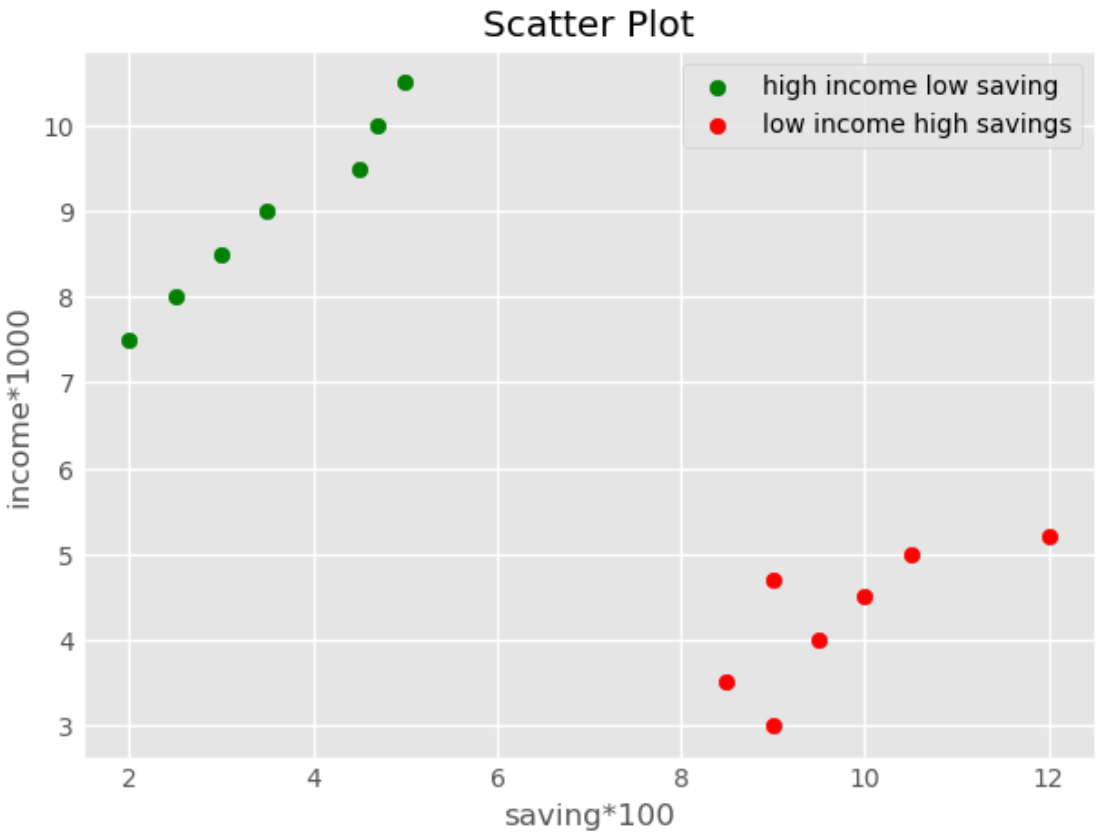
[26]:
```
from matplotlib import pyplot as plt
# Importing Numpy Library
import numpy as np
plt.style.use('fivethirtyeight')

mu = 50
sigma = 7
x = np.random.normal(mu, sigma, size=200)
fig, ax = plt.subplots()

ax.hist(x, 20)
ax.set_title('Historgram')
ax.set_xlabel('bin range')
ax.set_ylabel('frequency')

fig.tight_layout()
plt.show()
```



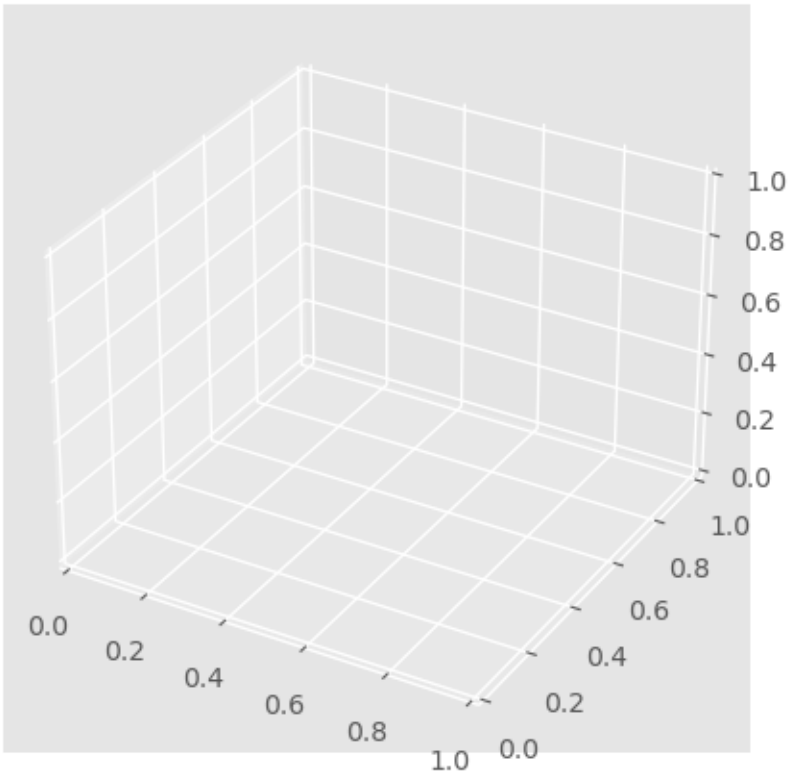
[27]:
```
from matplotlib import pyplot as plt
from matplotlib import style
style.use('ggplot')

x = [5,7,10]
y = [18,10,6]

x2 = [6,9,11]
y2 = [7,14,17]

plt.scatter(x, y)

plt.scatter(x2, y2, color='g')

plt.title('Epic Info')
plt.ylabel('Y axis')
plt.xlabel('X axis')

plt.show()
```



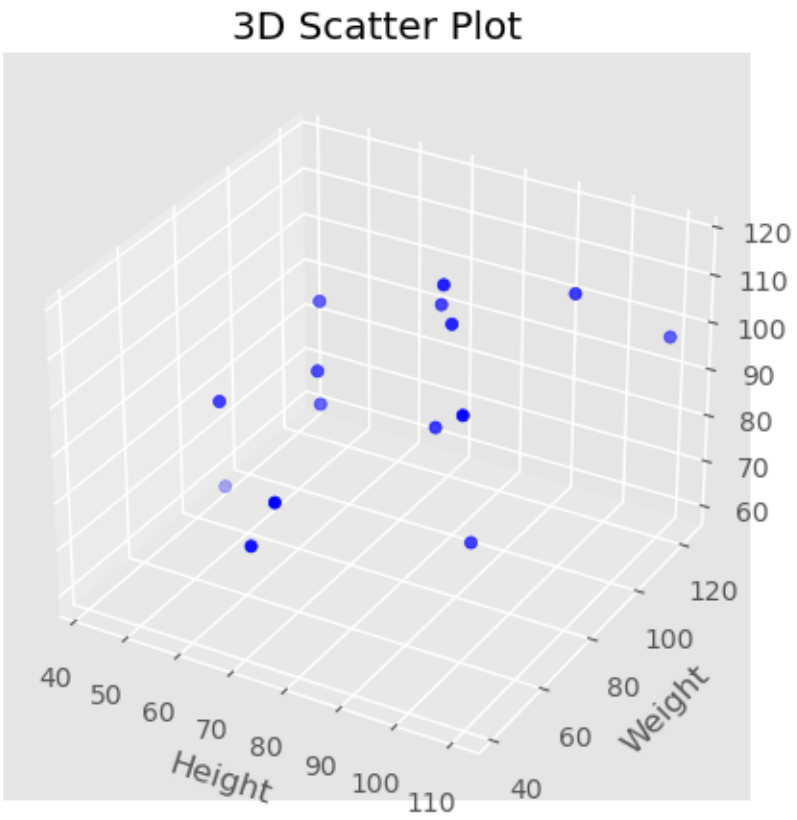
[28]:
```
import matplotlib.pyplot as plt
x = [2, 2.5, 3, 3.5, 4.5, 4.7, 5.0]
y = [7.5, 8, 8.5, 9, 9.5, 10, 10.5]

x1 = [9, 8.5, 9, 9.5, 10, 10.5, 12]
y1 = [3, 3.5, 4.7, 4, 4.5, 5, 5.2]
plt.scatter(x, y, label='high income low saving', color='g')
plt.scatter(x1, y1, label='low income high savings', color='r')
plt.xlabel('saving*100')
plt.ylabel('income*1000')
plt.title('Scatter Plot')
plt.legend()
plt.show()
```



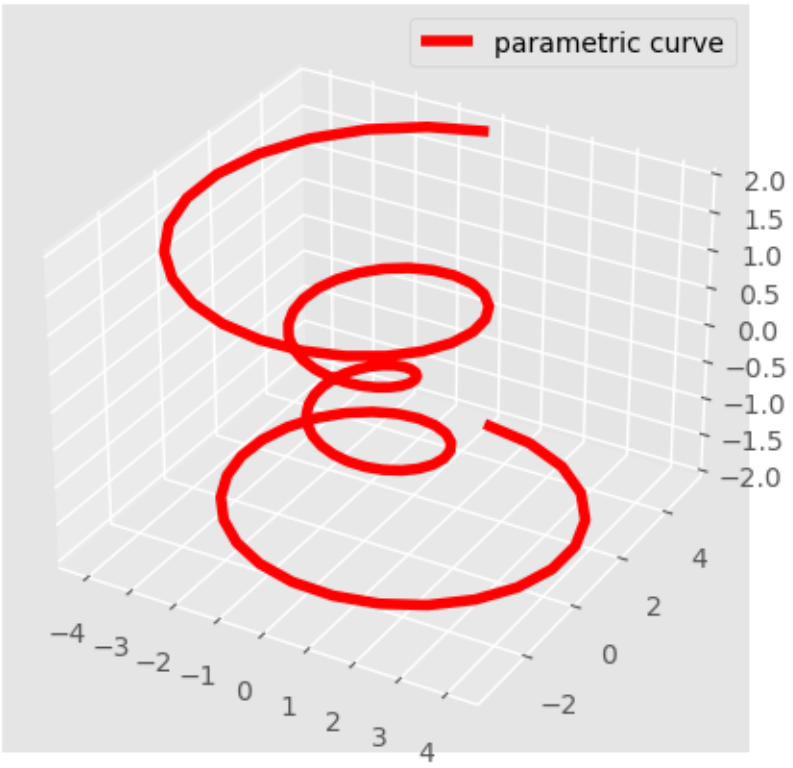
[36]:
```
from mpl_toolkits import mplot3d
import numpy as np
import matplotlib.pyplot as plt
fig = plt.figure()
ax = plt.axes(projection='3d')
```



[39]:
```
from mpl_toolkits import mplot3d
import numpy as np
import matplotlib.pyplot as plt

height = np.array([100, 110, 87, 85, 65, 80, 96, 75, 42, 59, 54, 63, 95, 71,␣
 ↪86])
weight = np.array([105, 123, 84, 85, 78, 95, 69, 42, 87, 91, 63, 83, 75, 41,␣
 ↪80])

# Create a random z-axis for demonstration (replace with your actual data)
z = np.random.randint(60, 120, size=len(height))

fig = plt.figure()
ax = plt.axes(projection='3d')

# 3D scatter plot with appropriate labels
ax.scatter3D(height, weight, z, color='blue')  # Add color for visual clarity
plt.title("3D Scatter Plot")
plt.xlabel("Height")
plt.ylabel("Weight")
plt.show()
```



[40]:
```
import matplotlib as mpl
from mpl_toolkits.mplot3d import Axes3D
import numpy as np
import matplotlib.pyplot as plt
mpl.rcParams['legend.fontsize'] = 10

fig = plt.figure()
ax = fig.add_subplot(projection='3d')
theta1 = np.linspace(-4 * np.pi, 4 * np.pi, 100)
z = np.linspace(-2, 2, 100)
r = z**2 + 1
x = r * np.sin(theta1)
y = r * np.cos(theta1)
ax.plot3D(x, y, z, label='parametric curve', color = 'red')
ax.legend()

plt.show()
```


parametric curve
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−4
−3
−2
−1
0
1
2
3
4
−2
0
2
4

[ ]:

and-write-with-csv-files-in-python

February 8, 2024

1 READ AND WRITE CSV FILES

[1]:
```
from platform import python_version

print(python_version())
```

```
3.11.5
```

[2]:
```
import csv
file = open('Salary_Data.csv')
type(file)
```

[2]:
```
_io.TextIOWrapper
```

[3]:
```
csvreader = csv.reader(file)
header = []
header = next(csvreader)
header
```

[3]:
```
['YearsExperience', 'Salary']
```

[5]:
```
rows = []
for row in csvreader:
 rows.append(row)
rows
```

[5]:
```
[['1.1', '39343.00'],
 ['1.3', '46205.00'],
 ['1.5', '37731.00'],
 ['2.0', '43525.00'],
 ['2.2', '39891.00'],
 ['2.9', '56642.00'],
 ['3.0', '60150.00'],
 ['3.2', '54445.00'],
 ['3.2', '64445.00'],
 ['3.7', '57189.00'],
 ['3.9', '63218.00'],
 ['4.0', '55794.00'],
```

```
 ['4.0', '56957.00'],
 ['4.1', '57081.00'],
 ['4.5', '61111.00'],
 ['4.9', '67938.00'],
 ['5.1', '66029.00'],
 ['5.3', '83088.00'],
 ['5.9', '81363.00'],
 ['6.0', '93940.00'],
 ['6.8', '91738.00'],
 ['7.1', '98273.00'],
 ['7.9', '101302.00'],
 ['8.2', '113812.00'],
 ['8.7', '109431.00'],
 ['9.0', '105582.00'],
 ['9.5', '116969.00'],
 ['9.6', '112635.00'],
 ['10.3', '122391.00'],
 ['10.5', '121872.00']]
```

[7]:
```
file.close()
```

[9]:
```
import csv
rows = []
with open("Salary_Data.csv", 'r') as file:
 csvreader = csv.reader(file)
 header = next(csvreader)
 for row in csvreader:
  rows.append(row)
print(header)
print(rows)
```

```
['YearsExperience', 'Salary']
[['1.1', '39343.00'], ['1.3', '46205.00'], ['1.5', '37731.00'], ['2.0',
'43525.00'], ['2.2', '39891.00'], ['2.9', '56642.00'], ['3.0', '60150.00'],
['3.2', '54445.00'], ['3.2', '64445.00'], ['3.7', '57189.00'], ['3.9',
'63218.00'], ['4.0', '55794.00'], ['4.0', '56957.00'], ['4.1', '57081.00'],
['4.5', '61111.00'], ['4.9', '67938.00'], ['5.1', '66029.00'], ['5.3',
'83088.00'], ['5.9', '81363.00'], ['6.0', '93940.00'], ['6.8', '91738.00'],
['7.1', '98273.00'], ['7.9', '101302.00'], ['8.2', '113812.00'], ['8.7',
'109431.00'], ['9.0', '105582.00'], ['9.5', '116969.00'], ['9.6', '112635.00'],
['10.3', '122391.00'], ['10.5', '121872.00']]
```

[10]:
```
with open('Salary_Data.csv') as file:
 content = file.readlines()
header = content[:1]
rows = content[1:]
print(header)
```

```
print(rows)
```

```
['YearsExperience,Salary\n']
['1.1,39343.00\n', '1.3,46205.00\n', '1.5,37731.00\n', '2.0,43525.00\n',
'2.2,39891.00\n', '2.9,56642.00\n', '3.0,60150.00\n', '3.2,54445.00\n',
'3.2,64445.00\n', '3.7,57189.00\n', '3.9,63218.00\n', '4.0,55794.00\n',
'4.0,56957.00\n', '4.1,57081.00\n', '4.5,61111.00\n', '4.9,67938.00\n',
'5.1,66029.00\n', '5.3,83088.00\n', '5.9,81363.00\n', '6.0,93940.00\n',
'6.8,91738.00\n', '7.1,98273.00\n', '7.9,101302.00\n', '8.2,113812.00\n',
'8.7,109431.00\n', '9.0,105582.00\n', '9.5,116969.00\n', '9.6,112635.00\n',
'10.3,122391.00\n', '10.5,121872.00']
```

[15]:
```
import pandas as pd

filename = "Salary_Data.csv"
data = pd.read_csv(filename, delimiter=',')
print(data)
```

```
    YearsExperience    Salary
0               1.1   39343.0
1               1.3   46205.0
2               1.5   37731.0
3               2.0   43525.0
4               2.2   39891.0
5               2.9   56642.0
6               3.0   60150.0
7               3.2   54445.0
8               3.2   64445.0
9               3.7   57189.0
10              3.9   63218.0
11              4.0   55794.0
12              4.0   56957.0
13              4.1   57081.0
14              4.5   61111.0
15              4.9   67938.0
16              5.1   66029.0
17              5.3   83088.0
18              5.9   81363.0
19              6.0   93940.0
20              6.8   91738.0
21              7.1   98273.0
22              7.9  101302.0
23              8.2  113812.0
24              8.7  109431.0
25              9.0  105582.0
26              9.5  116969.0
27              9.6  112635.0
28             10.3  122391.0
```

```
29              10.5  121872.0
```

[16]:
```
data.columns
```

[16]:
```
Index(['YearsExperience', 'Salary'], dtype='object')
```

[18]:
```
data.Salary
```

[18]:
```
0      39343.0
1      46205.0
2      37731.0
3      43525.0
4      39891.0
5      56642.0
6      60150.0
7      54445.0
8      64445.0
9      57189.0
10     63218.0
11     55794.0
12     56957.0
13     57081.0
14     61111.0
15     67938.0
16     66029.0
17     83088.0
18     81363.0
19     93940.0
20     91738.0
21     98273.0
22    101302.0
23    113812.0
24    109431.0
25    105582.0
26    116969.0
27    112635.0
28    122391.0
29    121872.0
Name: Salary, dtype: float64
```

[19]:
```
import csv
with open('Salary_Data.csv', 'r') as csvfile:
    reader = csv.DictReader(csvfile)
    print(row)
```

```
['10.5', '121872.00']
```

[20]:
```
import csv
filename = 'Students_Data.csv'
with open(filename, 'w', newline="") as file:
 csvwriter = csv.writer(file) # 2. create a csvwriter object
 csvwriter.writerow(header) # 4. write the header
 csvwriter.writerows(data) # 5. write the rest of the data
```

[24]:
```
header = ['Name', 'M1 Score', 'M2 Score']
data = [['Alex', 62, 80], ['Brad', 45, 56], ['Joey', 85, 98]]
filename = 'Student_scores.csv'
with open(filename, 'w') as file:
 for header in header:
  file.write(str(header)+', ')
 file.write('n')
 for row in data:
  for x in row:
   file.write(str(x)+', ')
  file.write('n')
```

[25]:
```
import pandas as pd
header = ['Name', 'M1 Score', 'M2 Score']
data = [['Alex', 62, 80], ['Brad', 45, 56], ['Joey', 85, 98]]
data = pd.DataFrame(data, columns=header)
```

[27]:
```
import csv
with open('Students_Data.csv', 'w', newline='') as csvfile:
    data = [{'Name': 'Alex', 'M1 Score': 62, 'M2 Score': 80},
 {'Name': 'Brad', 'M1 Score': 45, 'M2 Score': 56},
 {'Name': 'Joey', 'M1 Score': 85, 'M2 Score': 98}]
```

[ ]:
```

```