Name- Suryash pawar

Roll no.- 464

Batch- D4

Division- D

Program code-

```
import numpy as np
import csv
f1=open("D:\\dataset6.csv",'r')
f2=open("D:\\dataset6.csv",'r')
listcurrent=list(csv.reader(f1, delimiter=','))
listcurrent1=list(csv.reader(f2, delimiter=','))
d1=np.genfromtxt("C:\\Users\\Suryashpawar\\Desktop\\employees.csv",delimiter='
,',dtype=str)
listcurrent.pop(0)
listcurrent1.pop(0)
arr1=np.array(listcurrent)
arr = np.array([[11, 22, 33, 44, 55],
                [23, 24, 25, 26, 27],
                [12, 33, 11, 33, 55],
                [11, 22, 33, 44, 11],
                [103,444,666, 888,332]])
arr4 = np.array([[111, 212, 33, 434, 535],
                [232, 213, 235, 326, 327],
                [12, 33, 11, 4333, 553],
                [111, 22, 331, 44, 113],
                [1033,444,6663, 888,332]])
print("**********************performing the matrix operation
******************")

# Perform matrix operations
transpose = np.transpose(arr)
inverse = np.linalg.inv(arr)
determinant = np.linalg.det(arr)
transpose1=np.transpose(arr1)
inverse1=np.linalg.inv(arr)
determinant1=np.linalg.det(arr)
```

```
print("tanspose of the data set is= \n",transpose1)
print("inverse of the data set is= \n",inverse1)
print("determinant of the data set is =\n",determinant1)
print("Transpose of the matrix is:\n", transpose)
print("\nInverse of the matrix is:\n", inverse)
print("\nDeterminant of the matrix is:\n", determinant)
print(" **********performing the horizontal and vertical
stacking of the data**********")
arr2=np.array([[1],[2],[3],[4],[5]])
print("performing the horizonatl stack")
horizontal_stack = np.hstack((arr, arr2))
print(horizontal_stack)
print("performing the vertical stacking")
arr3=np.array([1,2,3,4,5])
vertical_stack=np.vstack((arr,arr3))
print(vertical_stack)
print("****performing the custom sequence generation****")
# Generate a sequence from 0 to 9
sequence = np.arange(10)
print(sequence)
print("Arithmetic and Statistical Operations, Mathematical Operations, Bitwise
Operators")
print(np.sum(arr,axis=1))
print("multiplying each element by 2")
print("original=\n",arr)
print("new")
addition=arr*2
print(addition)
print("substacting=\n")
sub=arr-2
print(sub)
# Statistical operations
print()
print("doing the statastical analysis=\n")
mean = np.mean(arr)
median = np.median(arr)
sum_all = np.sum(arr)
print(mean)
print(median)
print(sum_all)
# Mathematical operations
print()
print("doing the mathematical operation=\n")
square_root = np.sqrt(arr)
exponential = np.exp(arr)
logarithm = np.log(arr)
print(square_root)
print(exponential)
```

```
print(logarithm)
# Bitwise operators
print()
print("doing the bitwise opearion=\n")
bitwise_and = arr & 3
bitwise_or = arr | 3
bitwise_xor = arr ^ 3
print(bitwise_and)
print((bitwise_or))
print(bitwise_xor)
print("************creating the copy of the data=\n**************")
# Create a copy of the array
arr_copy = arr.copy()

# Create a view of the array
arr_view = arr.view()

# Modify the copy and view
arr_copy[0] = 10
arr_view[1] = 20

# Display the original array and its copies/views
print("Original array:", arr)
print("Copy of the array:", arr_copy)
print("View of the array:", arr_view)
print("*****Data Stacking, Searching, Sorting, Counting,
Broadcasting*********")
stacked_arr = np.stack((arr, arr4), axis=0)
print("stacked=\n")
print(stacked_arr)
print("#")
stacked_arr=np.stack((arr, arr4), axis=1)
print(stacked_arr)
print("searching =\n")
indices = np.where(arr > 3)
print(indices)
sorted_arr = np.sort(arr)
print("sorted array = \n",sorted_arr)
count = np.count_nonzero(arr == 55)
print("counting the number of variables =\n ",count)
print("braodcasting=\n")
# Perform broadcasting by adding arr2 to arr1
print("performing the broadcasting=\n")
result = arr + arr4
print("Result of broadcasting:")
print(result)
```

Output-

```
In [2]: runfile('C:/EDS_practical/lab_assignment/3__LAB.py', wdir='C:/EDS_practical/lab_assignment')
[['EMPLOYEE_ID' 'FIRST_NAME' 'LAST_NAME' 'EMAIL' 'PHONE_NUMBER'
  'HIRE_DATE' 'JOB_ID' 'SALARY' 'COMMISSION_PCT' 'MANAGER_ID'
  'DEPARTMENT_ID']
 ['198' 'Donald' 'OConnell' 'DOCONNEL' '650.507.9833' '21-Jun-07'
  'SH_CLERK' '2600' ' - ' '124' '50']
 ['199' 'Douglas' 'Grant' 'DGRANT' '650.507.9844' '13-Jan-08' 'SH_CLERK'
  '2600' ' - ' '124' '50']
 ['200' 'Jennifer' 'Whalen' 'JWHALEN' '515.123.4444' '17-Sep-03'
  'AD_ASST' '4400' ' - ' '101' '10']
 ['201' 'Michael' 'Hartstein' 'MHARTSTE' '515.123.5555' '17-Feb-04'
  'MK_MAN' '13000' ' - ' '100' '20']
 ['202' 'Pat' 'Fay' 'PFAY' '603.123.6666' '17-Aug-05' 'MK_REP' '6000'
  ' - ' '201' '20']
 ['203' 'Susan' 'Mavris' 'SMAVRIS' '515.123.7777' '07-Jun-02' 'HR_REP'
  '6500' ' - ' '101' '40']
 ['204' 'Hermann' 'Baer' 'HBAER' '515.123.8888' '07-Jun-02' 'PR_REP'
  '10000' ' - ' '101' '70']
 ['205' 'Shelley' 'Higgins' 'SHIGGINS' '515.123.8080' '07-Jun-02'
  'AC_MGR' '12008' ' - ' '101' '110']
 ['206' 'William' 'Gietz' 'WGIETZ' '515.123.8181' '07-Jun-02'
  'AC_ACCOUNT' '8300' ' - ' '205' '110']
 ['100' 'Steven' 'King' 'SKING' '515.123.4567' '17-Jun-03' 'AD_PRES'
  '24000' ' - ' ' - ' '90']
 ['101' 'Neena' 'Kochhar' 'NKOCHHAR' '515.123.4568' '21-Sep-05' 'AD_VP'
  '17000' ' - ' '100' '90']
 ['102' 'Lex' 'De Haan' 'LDEHAAN' '515.123.4569' '13-Jan-01' 'AD_VP'
  '17000' ' - ' '100' '90']
 ['103' 'Alexander' 'Hunold' 'AHUNOLD' '590.423.4567' '03-Jan-06'
  'IT_PROG' '9000' ' - ' '102' '60']
 ['104' 'Bruce' 'Ernst' 'BERNST' '590.423.4568' '21-May-07' 'IT_PROG'
  '6000' ' - ' '103' '60']
 ['105' 'David' 'Austin' 'DAUSTIN' '590.423.4569' '25-Jun-05' 'IT_PROG'
  '4800' ' - ' '103' '60']
 ['106' 'Valli' 'Pataballa' 'VPATABAL' '590.423.4560' '05-Feb-06'
  'IT_PROG' '4800' ' - ' '103' '60']
 ['107' 'Diana' 'Lorentz' 'DLORENTZ' '590.423.5567' '07-Feb-07' 'IT_PROG'
  '4200' ' - ' '103' '60']
 ['108' 'Nancy' 'Greenberg' 'NGREENBE' '515.124.4569' '17-Aug-02'
  'FI_MGR' '12008' ' - ' '101' '100']
```

```
  ' - ' '123' '50']
 ['140' 'Joshua' 'Patel' 'JPATEL' '650.121.1834' '06-Apr-06' 'ST_CLERK'
  '2500' ' - ' '123' '50']]
<class 'numpy.ndarray'>
<U14
['198' '199' '200' '201' '202' '203' '204' '205' '206' '100' '101' '102'
 '103' '104' '105' '106' '107' '108' '109' '110' '111' '112' '113' '114'
 '115' '116' '117' '118' '119' '120' '121' '122' '123' '124' '125' '126'
 '127' '128' '129' '130' '131' '132' '133' '134' '135' '136' '137' '138'
 '139' '140']
<class 'numpy.ndarray'>
6738
100
206
total number of records in the file is =  50
['JOB_ID' 'SH_CLERK' 'SH_CLERK' 'AD_ASST' 'MK_MAN' 'MK_REP' 'HR_REP'
 'PR_REP' 'AC_MGR' 'AC_ACCOUNT' 'AD_PRES' 'AD_VP' 'AD_VP' 'IT_PROG'
 'IT_PROG' 'IT_PROG' 'IT_PROG' 'IT_PROG' 'FI_MGR' 'FI_ACCOUNT'
 'FI_ACCOUNT' 'FI_ACCOUNT' 'FI_ACCOUNT' 'FI_ACCOUNT' 'PU_MAN' 'PU_CLERK'
 'PU_CLERK' 'PU_CLERK' 'PU_CLERK' 'PU_CLERK' 'ST_MAN' 'ST_MAN' 'ST_MAN'
 'ST_MAN' 'ST_MAN' 'ST_CLERK' 'ST_CLERK' 'ST_CLERK' 'ST_CLERK' 'ST_CLERK'
 'ST_CLERK' 'ST_CLERK' 'ST_CLERK' 'ST_CLERK' 'ST_CLERK' 'ST_CLERK'
 'ST_CLERK' 'ST_CLERK' 'ST_CLERK' 'ST_CLERK' 'ST_CLERK' 'ST_CLERK']
total number of records in the file is =  18
['2600' '2600' '4400' '13000' '6000' '6500' '10000' '12008' '8300' '24000'
 '17000' '17000' '9000' '6000' '4800' '4800' '4200' '12008' '9000' '8200'
 '7700' '7800' '6900' '11000' '3100' '2900' '2800' '2600' '2500' '8000'
 '8200' '7900' '6500' '5800' '3200' '2700' '2400' '2200' '3300' '2800'
 '2500' '2100' '3300' '2900' '2400' '2200' '3600' '3200' '2700' '2500']
index of the maximum== 9
the maximum salary is ==== 24000
the minimum salary index  is ==== 41
 the minimum salary is ==== 2100

doing the statastical analysis=

the mean of the salary is== 6182.32
the meadian of the salary is== 4600.0
sum of all the variable is == 309116
```

```
 ['127' 'James' 'Landry' 'JLANDRY' '650.124.1334' '14-Jan-07' 'ST_CLERK'
  '2400' ' - ' '120' '50']
 ['128' 'Steven' 'Markle' 'SMARKLE' '650.124.1434' '08-Mar-08' 'ST_CLERK'
  '2200' ' - ' '120' '50']
 ['129' 'Laura' 'Bissot' 'LBISSOT' '650.124.5234' '20-Aug-05' 'ST_CLERK'
  '3300' ' - ' '121' '50']
 ['130' 'Mozhe' 'Atkinson' 'MATKINSO' '650.124.6234' '30-Oct-05'
  'ST_CLERK' '2800' ' - ' '121' '50']
 ['131' 'James' 'Marlow' 'JAMRLOW' '650.124.7234' '16-Feb-05' 'ST_CLERK'
  '2500' ' - ' '121' '50']
 ['132' 'TJ' 'Olson' 'TJOLSON' '650.124.8234' '10-Apr-07' 'ST_CLERK'
  '2100' ' - ' '121' '50']
 ['133' 'Jason' 'Mallin' 'JMALLIN' '650.127.1934' '14-Jun-04' 'ST_CLERK'
  '3300' ' - ' '122' '50']
 ['134' 'Michael' 'Rogers' 'MROGERS' '650.127.1834' '26-Aug-06'
  'ST_CLERK' '2900' ' - ' '122' '50']
 ['135' 'Ki' 'Gee' 'KGEE' '650.127.1734' '12-Dec-07' 'ST_CLERK' '2400'
  ' - ' '122' '50']
 ['136' 'Hazel' 'Philtanker' 'HPHILTAN' '650.127.1634' '06-Feb-08'
  'ST_CLERK' '2200' ' - ' '122' '50']
 ['137' 'Renske' 'Ladwig' 'RLADWIG' '650.121.1234' '14-Jul-03' 'ST_CLERK'
  '3600' ' - ' '123' '50']
 ['138' 'Stephen' 'Stiles' 'SSTILES' '650.121.2034' '26-Oct-05'
  'ST_CLERK' '3200' ' - ' '123' '50']
 ['139' 'John' 'Seo' 'JSEO' '650.121.2019' '12-Feb-06' 'ST_CLERK' '2700'
  ' - ' '123' '50']
 ['140' 'Joshua' 'Patel' 'JPATEL' '650.121.1834' '06-Apr-06' 'ST_CLERK'
  '2500' ' - ' '123' '50']]
the sum of all st clerk salary is=
44000
```

```
the mean of the salary is== 6182.32
the meadian of the salary is== 4600.0
sum of all the variable is == 309116
['Donald' 'Douglas' 'Jennifer' 'Michael' 'Pat' 'Susan' 'Hermann' 'Shelley'
 'William' 'Steven' 'Neena' 'Lex' 'Alexander' 'Bruce' 'David' 'Valli'
 'Diana' 'Nancy' 'Daniel' 'John' 'Ismael' 'Jose Manuel' 'Luis' 'Den'
 'Alexander' 'Shelli' 'Sigal' 'Guy' 'Karen' 'Matthew' 'Adam' 'Payam'
 'Shanta' 'Kevin' 'Julia' 'Irene' 'James' 'Steven' 'Laura' 'Mozhe' 'James'
 'TJ' 'Jason' 'Michael' 'Ki' 'Hazel' 'Renske' 'Stephen' 'John' 'Joshua']
highest paid employ is== Steven 24000 AC_ACCOUNT
lowest paid employ is== TJ 2100 ST_CLERK
disploying all the records of the highest paid is
['100' 'Steven' 'King' 'SKING' '515.123.4567' '17-Jun-03' 'AD_PRES'
 '24000' ' - ' ' - ' '90']
disploying all the records of the lowest paid is
['132' 'TJ' 'Olson' 'TJOLSON' '650.124.8234' '10-Apr-07' 'ST_CLERK' '2100'
 ' - ' '121' '50']
printing all the st clerks
[['125' 'Julia' 'Nayer' 'JNAYER' '650.124.1214' '16-Jul-05' 'ST_CLERK'
  '3200' ' - ' '120' '50']
 ['126' 'Irene' 'Mikkilineni' 'IMIKKILI' '650.124.1224' '28-Sep-06'
  'ST_CLERK' '2700' ' - ' '120' '50']
 ['127' 'James' 'Landry' 'JLANDRY' '650.124.1334' '14-Jan-07' 'ST_CLERK'
  '2400' ' - ' '120' '50']
 ['128' 'Steven' 'Markle' 'SMARKLE' '650.124.1434' '08-Mar-08' 'ST_CLERK'
  '2200' ' - ' '120' '50']
 ['129' 'Laura' 'Bissot' 'LBISSOT' '650.124.5234' '20-Aug-05' 'ST_CLERK'
  '3300' ' - ' '121' '50']
 ['130' 'Mozhe' 'Atkinson' 'MATKINSO' '650.124.6234' '30-Oct-05'
  'ST_CLERK' '2800' ' - ' '121' '50']
 ['131' 'James' 'Marlow' 'JAMRLOW' '650.124.7234' '16-Feb-05' 'ST_CLERK'
  '2500' ' - ' '121' '50']
 ['132' 'TJ' 'Olson' 'TJOLSON' '650.124.8234' '10-Apr-07' 'ST_CLERK'
  '2100' ' - ' '121' '50']
 ['133' 'Jason' 'Mallin' 'JMALLIN' '650.127.1934' '14-Jun-04' 'ST_CLERK'
  '3300' ' - ' '122' '50']
 ['134' 'Michael' 'Rogers' 'MROGERS' '650.127.1834' '26-Aug-06'
  'ST_CLERK' '2900' ' - ' '122' '50']
```