DASARI.NAGAVENI

EMAIL: dasari.nagaveni2020@vitstudent.ac.in

Vellore institute of technology

Chennai

Assignment-1

Assignment 1

1. Assign your Name to variable name and Age to variable age. Make a Python program that prints your name and age.

2. X="Datascience is used to extract meaningful insights."

Split the string

3. Make a function that gives multiplication of two numbers

4. Create a Dictionary of 5 States with their capitals. also print the keys and values.

5. Create a list of 1000 numbers using range function.

6. Create an identity matrix of dimension 4 by 4

7. Create a 3x3 matrix with values ranging from 1 to 9

8. Create 2 similar dimensional array and perform sum on them.

9. Generate the series of dates from 1st Feb, 2023 to 1st March, 2023 (both inclusive)

10. Given a dictionary, convert it into corresponding dataframe and display it

dictionary = {'Brand': ['Maruti', 'Renault', 'Hyndai'], 'Sales' : [250, 200, 240]}

[1]

QUESTION : 1. Assign your Name to variable name and Age to variable age. Make a Python program that prints your name and age.

CODE

```
name = "DASARI.NAGAVENI"

age = "21"



print("Name:", name)

print("Age:", age)
```

OUTPUT

```
Age: 21
```

[2]

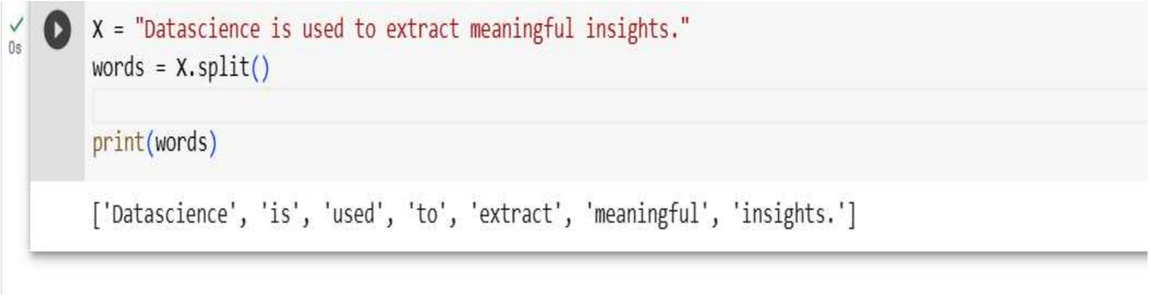
QUESTION : X="Datascience is used to extract meaningful insights."

Split the string

CODE

```
X = "Datascience is used to extract meaningful insights."

words = X.split()



print(words)
```

OUTPUT



```
X = "Datascience is used to extract meaningful insights."
words = X.split()

print(words)
```

```
['Datascience', 'is', 'used', 'to', 'extract', 'meaningful', 'insights.']
```

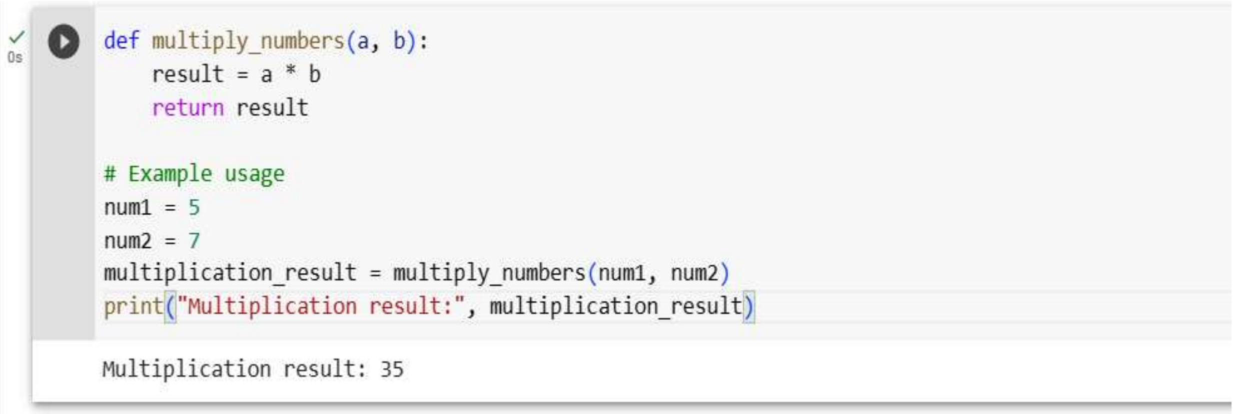
[3]

QUESTION : Make a function that gives multiplication of two numbers

CODE :

```
def multiply_numbers(a, b):
    result = a * b
    return result



# Example usage
num1 = 5
num2 = 7
multiplication_result = multiply_numbers(num1, num2)
print("Multiplication result:", multiplication_result)
```

OUTPUT



```
def multiply_numbers(a, b):
    result = a * b
    return result

# Example usage
num1 = 5
num2 = 7
multiplication_result = multiply_numbers(num1, num2)
print("Multiplication result:", multiplication_result)
```

```
Multiplication result: 35
```

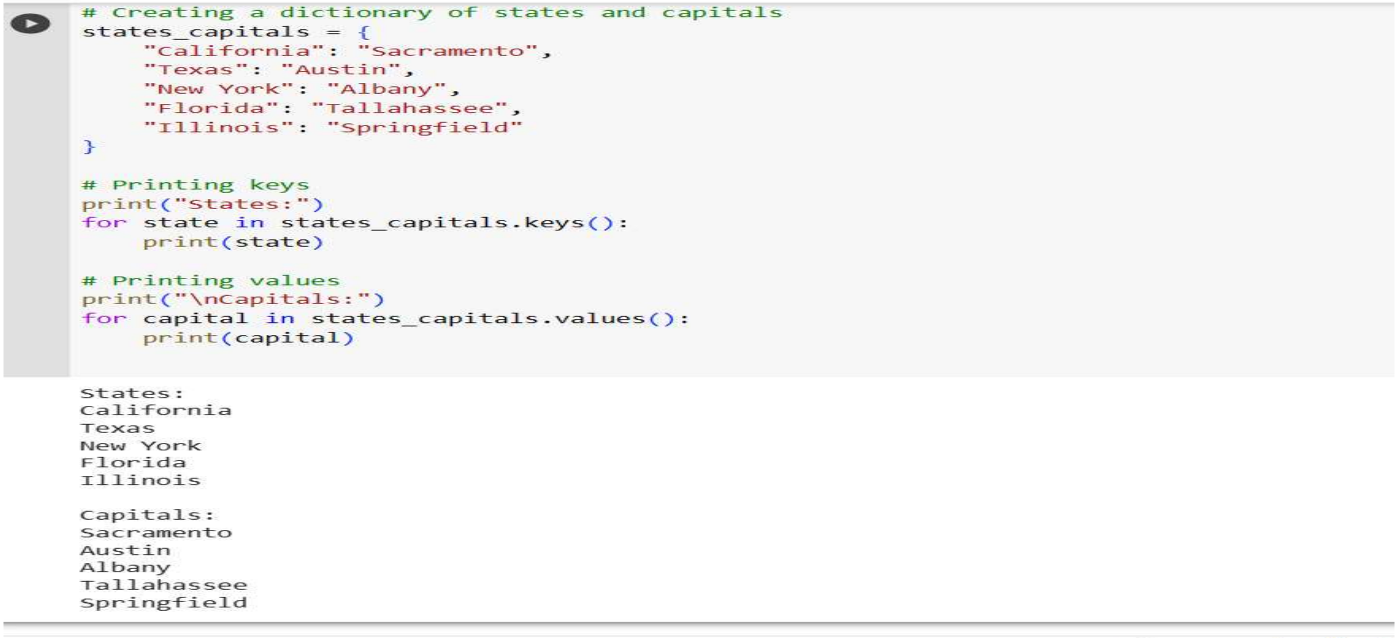
[4]

QUESTION : Create a Dictionary of 5 States with their capitals. also print the keys and values.

CODE

```
# Creating a dictionary of states and capitals
states_capitals = {
    "California": "Sacramento",
    "Texas": "Austin",
    "New York": "Albany",
    "Florida": "Tallahassee",

    "Illinois": "Springfield"

}



# Printing keys

print("States:")

for state in states_capitals.keys():

    print(state)



# Printing values

print("\nCapitals:")

for capital in states_capitals.values():

    print(capital)
```

OUTPUT



```
# Creating a dictionary of states and capitals
states_capitals = {
    "California": "Sacramento",
    "Texas": "Austin",
    "New York": "Albany",
    "Florida": "Tallahassee",
    "Illinois": "Springfield"
}

# Printing keys
print("States:")
for state in states_capitals.keys():
    print(state)

# Printing values
print("\nCapitals:")
for capital in states_capitals.values():
    print(capital)
```

```
States:
California
Texas
New York
Florida
Illinois

Capitals:
Sacramento
Austin
Albany
Tallahassee
Springfield
```

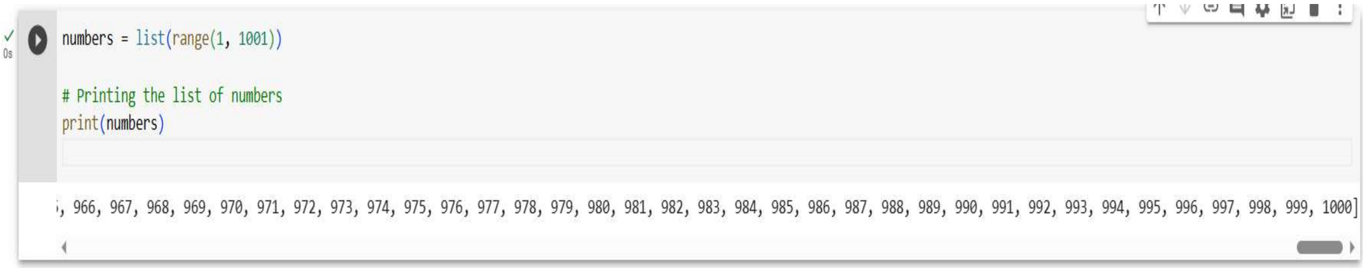
[5]

QUESTION : Create a list of 1000 numbers using range function.

CODE

```
numbers = list(range(1, 1001))


# Printing the list of numbers

print(numbers)
```

OUTPUT



```
numbers = list(range(1, 1001))

# Printing the list of numbers
print(numbers)
```

```
, 966, 967, 968, 969, 970, 971, 972, 973, 974, 975, 976, 977, 978, 979, 980, 981, 982, 983, 984, 985, 986, 987, 988, 989, 990, 991, 992, 993, 994, 995, 996, 997, 998, 999, 1000]
```

[6]

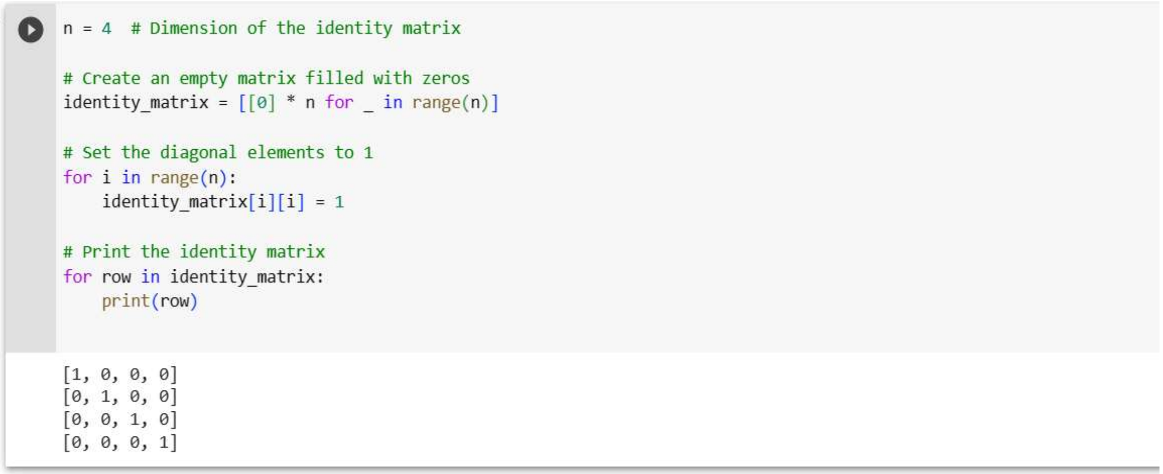
QUESTION : Create an identity matrix of dimension 4 by 4

CODE

```
n = 4  # Dimension of the identity matrix


# Create an empty matrix filled with zeros

identity_matrix = [[0] * n for _ in range(n)]


# Set the diagonal elements to 1

for i in range(n):

    identity_matrix[i][i] = 1
```

```
# Print the identity matrix

for row in identity_matrix:

    print(row)
```

OUTPUT



```
n = 4  # Dimension of the identity matrix

# Create an empty matrix filled with zeros
identity_matrix = [[0] * n for _ in range(n)]

# Set the diagonal elements to 1
for i in range(n):
    identity_matrix[i][i] = 1

# Print the identity matrix
for row in identity_matrix:
    print(row)
```

```
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 1, 0]
[0, 0, 0, 1]
```

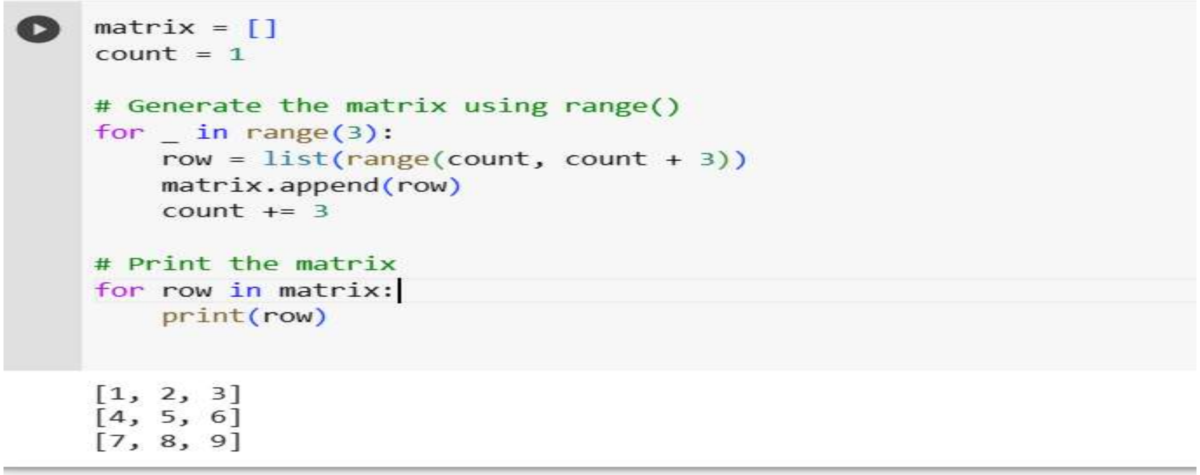
[**[7]**

QUESTION : Create a 3x3 matrix with values ranging from 1 to 9

CODE

```
matrix = []

count = 1



# Generate the matrix using range()

for _ in range(3):

    row = list(range(count, count + 3))

    matrix.append(row)

    count += 3



# Print the matrix

for row in matrix:
    print(row)
```

OUTPUT



```
matrix = []
count = 1

# Generate the matrix using range()
for _ in range(3):
    row = list(range(count, count + 3))
    matrix.append(row)
    count += 3

# Print the matrix
for row in matrix:
    print(row)
```

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

[8]

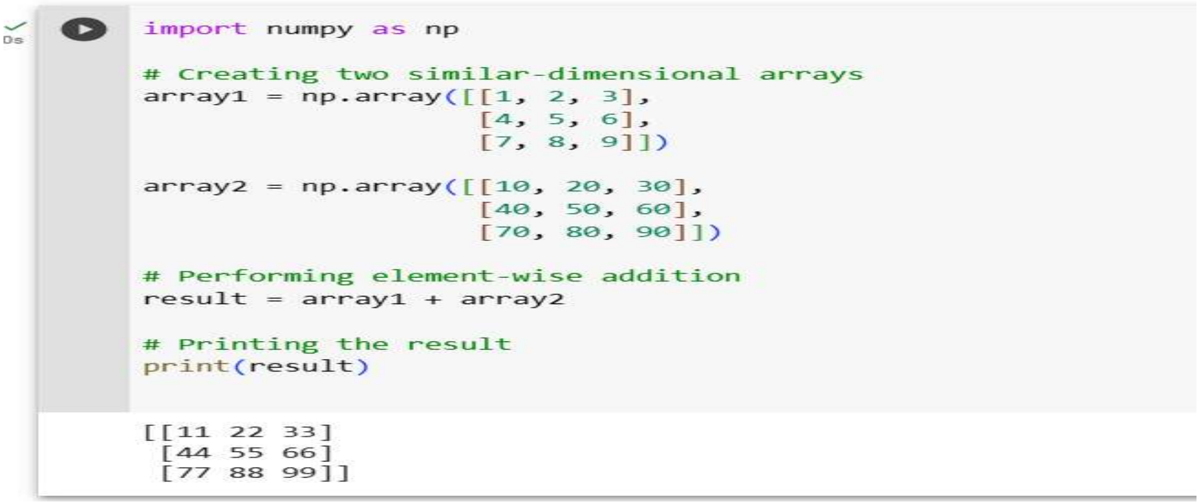
QUESTION : Create 2 similar dimensional array and perform sum on them.

CODE

```
import numpy as np



# Creating two similar-dimensional arrays

array1 = np.array([[1, 2, 3],

              [4, 5, 6],

              [7, 8, 9]])



array2 = np.array([[10, 20, 30],

              [40, 50, 60],

              [70, 80, 90]])
```

```
# Performing element-wise addition

result = array1 + array2



# Printing the result

print(result)
```

OUTPUT



```
import numpy as np

# Creating two similar-dimensional arrays
array1 = np.array([[1, 2, 3],
                   [4, 5, 6],
                   [7, 8, 9]])

array2 = np.array([[10, 20, 30],
                   [40, 50, 60],
                   [70, 80, 90]])

# Performing element-wise addition
result = array1 + array2

# Printing the result
print(result)
```

```
[[11 22 33]
 [44 55 66]
 [77 88 99]]
```

[9]

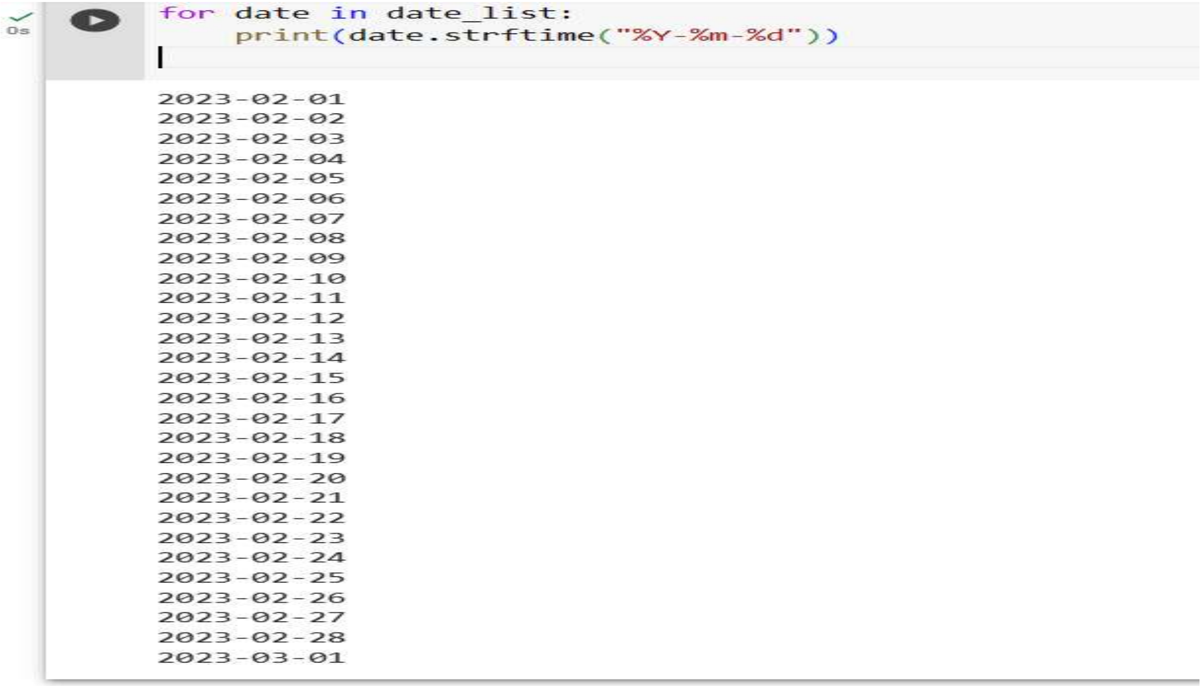
QUESTION : Generate the series of dates from 1st Feb, 2023 to 1st March, 2023 (both inclusive)

CODE

```
from datetime import datetime, timedelta



start_date = datetime(2023, 2, 1)

end_date = datetime(2023, 3, 1)



date_list = []
```

```
current_date = start_date



while current_date <= end_date:

    date_list.append(current_date)

    current_date += timedelta(days=1)



# Print the list of dates

for date in date_list:

    print(date.strftime("%Y-%m-%d"))
```

OUTPUT



```
for date in date_list:
    print(date.strftime("%Y-%m-%d"))
```

```
2023-02-01
2023-02-02
2023-02-03
2023-02-04
2023-02-05
2023-02-06
2023-02-07
2023-02-08
2023-02-09
2023-02-10
2023-02-11
2023-02-12
2023-02-13
2023-02-14
2023-02-15
2023-02-16
2023-02-17
2023-02-18
2023-02-19
2023-02-20
2023-02-21
2023-02-22
2023-02-23
2023-02-24
2023-02-25
2023-02-26
2023-02-27
2023-02-28
2023-03-01
```

[10]

QUESTION Given a dictionary, convert it into corresponding dataframe and display it

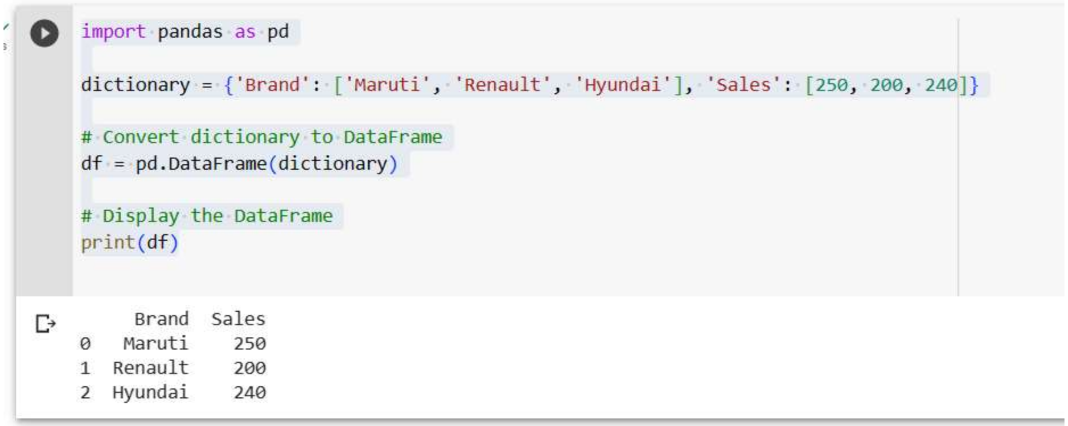
dictionary = {'Brand': ['Maruti', 'Renault', 'Hyndai'], 'Sales' : [250, 200, 240]}

CODE

```
import pandas as pd
```

```
dictionary = {'Brand': ['Maruti', 'Renault', 'Hyundai'], 'Sales': [250, 200, 240]}


# Convert dictionary to DataFrame

df = pd.DataFrame(dictionary)


# Display the DataFrame

print(df)
```

OUTPUT



```
import pandas as pd

dictionary = {'Brand': ['Maruti', 'Renault', 'Hyundai'], 'Sales': [250, 200, 240]}

# Convert dictionary to DataFrame
df = pd.DataFrame(dictionary)

# Display the DataFrame
print(df)
```

```
     Brand  Sales
0   Maruti    250
1  Renault    200
2  Hyundai    240
```