SMART BRIDGE EXTERNSHIP IN

APPLIED DATA SCIENCE

ASSIGNMENT 1

NAME : JEYASRI VARTHINI B

REG.NO. : 20MID0049

BRANCH : Integrated M.Tech. Computer Science with specialization in Data Science

CAMPUS : VIT VELLORE

EMAIL : jeyasrivarthini.b2020@vitstudent.ac.in

DATE : 19-05-2023

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

In [1]:
```
name = "JEYASRI VARTHINI B"
age = 20
print(name,age)
```

```
JEYASRI VARTHINI B 20
```

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

In [2]:
```
x = "Datascience is used to extract meaningful insights."
x_split = x.split()
print(x_split)
```

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

3. Make a function that gives multiplication of two numbers.

In [3]:
```
def multiply (n1,n2):
    return n1 * n2
print(multiply(3,4))
print(multiply(3.1,4.2))
```

```
12
13.020000000000001
```

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

In [4]:
```
states_capitals = {"Tamil Nadu":"Chennai", "Kerala":"Thiruvananthapuram", "Karnataka":"Bangalore",
                   "Maharashtra":"Mumbai", "Himachal Pradesh":"Shimla"}
print("Keys: ",states_capitals.keys())
print("Values: ",states_capitals.values())
print("Keys and Values: ",states_capitals.items())
```

```
Keys:  dict_keys(['Tamil Nadu', 'Kerala', 'Karnataka', 'Maharashtra', 'Himachal Pradesh'])
Values:  dict_values(['Chennai', 'Thiruvananthapuram', 'Bangalore', 'Mumbai', 'Shimla'])
Keys and Values:  dict_items([('Tamil Nadu', 'Chennai'), ('Kerala', 'Thiruvananthapuram'), ('Karnatak
a', 'Bangalore'), ('Maharashtra', 'Mumbai'), ('Himachal Pradesh', 'Shimla')])
```

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

In [5]:
```
list_1000 = [x for x in range(1000)]
print(list_1000)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 2
7, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 5
2, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 7
7, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101,
102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 1
22, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 14
2, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 16
2, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 18
2, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 20
2, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 22
2, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 24
2, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 26
2, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 28
2, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 30
2, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 32
2, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 34
2, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 36
2, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 38
2, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 40
2, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 42
2, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 44
2, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 46
2, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 48
2, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 50
```

```
2, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 54
2, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 56
2, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 58
2, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598, 599, 600, 601, 60
2, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 62
2, 623, 624, 625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 64
2, 643, 644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 66
2, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675, 676, 677, 678, 679, 680, 681, 68
2, 683, 684, 685, 686, 687, 688, 689, 690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 70
2, 703, 704, 705, 706, 707, 708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 72
2, 723, 724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741, 74
2, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755, 756, 757, 758, 759, 760, 761, 76
2, 763, 764, 765, 766, 767, 768, 769, 770, 771, 772, 773, 774, 775, 776, 777, 778, 779, 780, 781, 78
2, 783, 784, 785, 786, 787, 788, 789, 790, 791, 792, 793, 794, 795, 796, 797, 798, 799, 800, 801, 80
2, 803, 804, 805, 806, 807, 808, 809, 810, 811, 812, 813, 814, 815, 816, 817, 818, 819, 820, 821, 82
2, 823, 824, 825, 826, 827, 828, 829, 830, 831, 832, 833, 834, 835, 836, 837, 838, 839, 840, 841, 84
2, 843, 844, 845, 846, 847, 848, 849, 850, 851, 852, 853, 854, 855, 856, 857, 858, 859, 860, 861, 86
2, 863, 864, 865, 866, 867, 868, 869, 870, 871, 872, 873, 874, 875, 876, 877, 878, 879, 880, 881, 88
2, 883, 884, 885, 886, 887, 888, 889, 890, 891, 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, 90
2, 903, 904, 905, 906, 907, 908, 909, 910, 911, 912, 913, 914, 915, 916, 917, 918, 919, 920, 921, 92
2, 923, 924, 925, 926, 927, 928, 929, 930, 931, 932, 933, 934, 935, 936, 937, 938, 939, 940, 941, 94
2, 943, 944, 945, 946, 947, 948, 949, 950, 951, 952, 953, 954, 955, 956, 957, 958, 959, 960, 961, 96
2, 963, 964, 965, 966, 967, 968, 969, 970, 971, 972, 973, 974, 975, 976, 977, 978, 979, 980, 981, 98
2, 983, 984, 985, 986, 987, 988, 989, 990, 991, 992, 993, 994, 995, 996, 997, 998, 999]
```

6. Create an identity matrix of dimension 4 by 4.

In [6]:
```
import numpy as np
eye_4 = np.eye(4)
print(eye_4)
```

```
[[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
```

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

In [7]:
```
import numpy as np
mat = np.arange(1,10).reshape(3,3)
print(mat)
```

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

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

In [8]:
```
import numpy as np
arr1 = np.arange(1,10).reshape(3,3)
arr2 = np.arange(11,20).reshape(3,3)
sum_arr = arr1 + arr2
print(sum_arr)
```

```
[[12 14 16]
 [18 20 22]
 [24 26 28]]
```

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

In [9]:
```
import pandas as pd
from datetime import datetime
date_series = pd.Series(pd.date_range(start="2023-02-01",end="2023-03-01"))
print(date_series)
```

```
0    2023-02-01
1    2023-02-02
2    2023-02-03
3    2023-02-04
4    2023-02-05
5    2023-02-06
6    2023-02-07
7    2023-02-08
8    2023-02-09
9    2023-02-10
10   2023-02-11
```

10. Given a dictionary, convert it into corresponding dataframe and display it: dictionary = {'Brand': ['Maruti', 'Renault', 'Hyndai'], 'Sales' : [250, 200, 240]}

In [10]:
```
import pandas as pd
dictionary = {'Brand': ['Maruti', 'Renault', 'Hyndai'], 'Sales': [250, 200, 240]}
df = pd.DataFrame(dictionary)
df
```

Out[10]:

	Brand	Sales
0	Maruti	250
1	Renault	200
2	Hyndai	240

```
11   2023-02-12
12   2023-02-13
13   2023-02-14
14   2023-02-15
15   2023-02-16
16   2023-02-17
17   2023-02-18
18   2023-02-19
19   2023-02-20
20   2023-02-21
21   2023-02-22
22   2023-02-23
23   2023-02-24
24   2023-02-25
25   2023-02-26
26   2023-02-27
27   2023-02-28
28   2023-03-01
dtype: datetime64[ns]
```

JUPITER NOTEBOOK (.ipynb file) IN GITHUB REPOSITORY:

https://github.com/JeyasriVarthiniB/Smart-Bridge-Externship-in-Applied-Data-Science/blob/main/JEYASRI_VARTHINI_B_ADS_ASSIGNMENT_1.ipynb