Python Use Cases

Use case 1

1. Ask user information, for ex: name, department, college etc and display them in the following format

---

Name : learnbyexample
Department : ECE
College : PSG Tech

```
In [2]: name = input("Please enter the NAME of the Student: ")
        dept= input("Please enter the Student's Department: ")
        col = input("Please enter the College name of the Student: ")
        print("Name: ", name)
        print("Department: ", dept)
        print("College: ", col)

          Please enter the NAME of the Student: Sundaresan
          Please enter the Student's Department: EEE
          Please enter the College name of the Student: BIST
          Name:  Sundaresan
          Department:  EEE
          College:  BIST
```

```
1. name = input("Please enter the NAME of the Student: ")
2. dept= input("Please enter the Student's Department: ")
3. col = input("Please enter the College name of the Student: ")
4. print("Name: ", name)
5. print("Department: ", dept)
6. print("College: ", col)
```

Use Case 2

2. Get Sum of 20 and 30

```
In [6]: # Get the numbers
        list = (20,30)
        # sum(list or numbers)
        mysum = sum(list)
        #print the variable mysum
        print("Sum of 20 and 30 is:", mysum)

          Sum of 20 and 30 is: 50
```

```
1. # Get the numbers
2. list = (20,30)
3. # sum(list or numbers)
4. mysum = sum(list)
5. #print the variable mysum
6. print("Sum of 20 and 30 is:", mysum)
```

Use Case 3

3. Find root 2 and square 2

In [33]:
```
# import the math module
import math

# enter the number
nbr = int(input ("Enter the number: "))

# print the square root of Number
sqrt=float(math.sqrt(nbr))
print("Square root of the number "+str(nbr)+" is "+str(round (sqrt,3)))

# print the square of Number
square=float(math.pow(nbr,2))
print("Square of the number "+str(nbr)+" is "+str(round (square,3)))
```

```
Enter the number: 2
Square root of the number 2 is 1.414
Square of the number 2 is 4.0
```

```
1.  # import the math module
2.  import math
3.
4.  # enter the number
5.  nbr = int(input ("Enter the number: "))
6.
7.  # print the square root of Number
8.  sqrt=float(math.sqrt(nbr))
9.  print("Square root of the number "+str(nbr)+" is "+str(round (sqrt,3)))
10.
11. # print the square of Number
12. square=float(math.pow(nbr,2))
13. print("Square of the number "+str(nbr)+" is "+str(round (square,3)))
```

Use Case 4

4. Write a Python program that accepts an integer (n) and print n+nn+nnn

In [38]:
```
# Get the integer value
nbr = int(input("Enter an integer number : "))
a1 = int( "%s" % nbr )
a2 = int( "%s%s" % (nbr,nbr) )
a3 = int( "%s%s%s" % (nbr,nbr,nbr) )
print ("Value of "+str(nbr)+"+"+str(nbr)+str(nbr)+"+"+str(nbr)+str(nbr)+str(nbr)+" is "+str(a1+a2+a3))
```

```
Enter an integer number : 6
Value of 6+66+666 is 738
```

```
1.  # Get the integer value
2.  nbr = int(input("Enter an integer number : "))
3.  a1 = int( "%s" % nbr )
4.  a2 = int( "%s%s" % (nbr,nbr) )
5.  a3 = int( "%s%s%s" % (nbr,nbr,nbr) )
6.  print ("Value of "+str(nbr)+"+"+str(nbr)+str(nbr)+"+"+str(nbr)+str(nbr)+str(nbr)+" is "
    +str(a1+a2+a3))
```

Use Case 5

5. Print first and last characters of Hello using indexing

```
In [46]: # Store the string
         str="Hello"
         # Print First char of the string
         print ("First char of the string "+str+" is "+str[0])
         # Print LAst char of the string
         print ("Last char of the string "+str+" is "+str[len(str)-1])

         First char of the string Hello is H
         Last char of the string Hello is o
```

```
1. # Store the string
2. str="Hello"
3. # Print First char of the string
4. print ("First char of the string "+str+" is "+str[0])
5. # Print Last char of the string
6. print ("Last char of the string "+str+" is "+str[len(str)-1])
```

Use Case 6

6. Print Hello in reverse using slicing

```
In [25]: # Store the string
         str="Hello"
         # Print string in reverse
         print ("Reversed String of "+str+" is "+str[::-1])

         Reversed String of Hello is olleH
```

```
1. #Store the string
2. str = "Hello"
3. #Print string in reverse
4. print ("Reversed String of " + str + " is " + str[::-1])
```

Use Case 7

7. Build list [0,0,0] in two seperate ways

```
In [37]: # Print a list of 3 Zeros - Method 1
         print ([0] * 3)
         # Print a list of 3 Zeros - Method 2
         str = [0 for i in range(3)]
         print (str)

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

```
# Print a list of 3 Zeros - Method 1
print ([0] * 3)
# Print a list of 3 Zeros - Method 2
str = [0 for i in range(3)]
print (str)
```

Use Case 8

8. Change value of hello to goodbye

```
In [43]: # Finding and Replacing Element in List
         listx=[1,2,3,4,"hello"]
         listx[4]="goodbye"
         print (listx)

          [1, 2, 3, 4, 'goodbye']
```

```
# Finding and Replacing Element in List
listx=[1,2,3,4,"hello"]
listx[4]="goodbye"
print (listx)
```

Use Case 9

9. Sort the list below

```
In [47]: # Sorting Numbers
         listx=[4,5,1,3,2]
         listx.sort()
         print ("Sorted String is ")
         print (listx)

          Sorted String is
          [1, 2, 3, 4, 5]
```

```
# Sorting Numbers
listx=[4,5,1,3,2]
listx.sort()
print ("Sorted String is ")
print (listx)
```

Use Case 10

Question Unclear

Use Case 11

11. Find unique values from list below

```
In [1]: # Finding Unique values in the list
        listx=[1,2,2,33,4,4,11,22,3,3,2]
        listy=set(listx)
        print ("Unique values in listx is ")
        print (listy)

        Unique values in listx is
        {1, 2, 33, 4, 3, 11, 22}
```

```
1. # Finding Unique values in the list
2. listx=[1,2,2,33,4,4,11,22,3,3,2]
3. listy=set(listx)
4. print ("Unique values in listx is ")
5. print (listy)
```

Use Case 12

12. Write a program which will find all such numbers which are divisible by 7 but are not a multiple of 5, between 2000 and 3200 (both included).

```
In [4]: # Find all numbers divisible by 7 and not multiples of 5 between 2000 and 3200
        listx=[]
        for nbr in range(2000, 3200):
            if (nbr%7==0) and (nbr%5!=0):
                listx.append(str(nbr))
        print (';'.join(listx))

        2002;2009;2016;2023;2037;2044;2051;2058;2072;2079;2086;2093;2107;2114;2121;2128;2142;2149;2156;2163;2177;2184;2191;21
        98;2212;2219;2226;2233;2247;2254;2261;2268;2282;2289;2296;2303;2317;2324;2331;2338;2352;2359;2366;2373;2387;2394;240
        1;2408;2422;2429;2436;2443;2457;2464;2471;2478;2492;2499;2506;2513;2527;2534;2541;2548;2562;2569;2576;2583;2597;2604;
        2611;2618;2632;2639;2646;2653;2667;2674;2681;2688;2702;2709;2716;2723;2737;2744;2751;2758;2772;2779;2786;2793;2807;28
        14;2821;2828;2842;2849;2856;2863;2877;2884;2891;2898;2912;2919;2926;2933;2947;2954;2961;2968;2982;2989;2996;3003;301
        7;3024;3031;3038;3052;3059;3066;3073;3087;3094;3101;3108;3122;3129;3136;3143;3157;3164;3171;3178;3192;3199
```

```
1. # Find all numbers divisible by 7 and not multiples of 5 between 2000 and 3200
2. listx=[]
3. for nbr in range(2000, 3200):
4.     if (nbr%7==0) and (nbr%5!=0):
5.         listx.append(str(nbr))
6. print (';'.join(listx))
```

Use Case 13

13. Print the following pattern using recursion

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

In [22]:

```
# Print Number pattern

# Function definition for printing the row
def print_myrow(a, b):
    if (a == 0):
        return;
    print(b , end=" ");

    # Calling Recursive print_myrow()
    print_myrow(a - 1, b);

# Function definition for printing the pattern
def print_mypattern(x, nbr):
    if (x == 0):
        return;
    print_myrow(nbr - x + 1, nbr - x + 1);
    print("");

    # Calling Recursive print_mypattern()
    print_mypattern(x - 1, nbr);

# Main
val = 5;
print_mypattern(val, val);
```

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

```
# Print Number pattern

# Function definition for printing the row
def print_myrow(a, b):
    if (a == 0):
      return;
    print(b , end=" ");

    # Calling Recursive print_myrow()
    print_myrow(a - 1, b);

# Function definition for printing the pattern
def print_mypattern(x, nbr):
    if (x == 0):
      return;
    print_myrow(nbr - x + 1, nbr - x + 1);
    print("");

    # Calling Recursive print_mypattern()
    print_mypattern(x - 1, nbr);

# Main
val = 5;
print_mypattern(val, val);
```

Use Case 14

Working on

Use Case 15

15. Get a Sentence from user and print all words with even number of letters

In [6]:
```
# Display Even length words in a Sentence

# Get String from User
str = input("Please write a Sentence: ")

# Move the sentence into a List
listx = list(str.split(' '))

print ("Your sentence is: ", str);

# Converted list of words
print ("List string is : ", listx);

# Check word by word get its length and print only when the word is EVEN
print ("EVEN Length Words are:");
for word in listx:
    if(len(word)%2==0 ):
        print (word);
```

```
Please write a Sentence: This can be the test sentence
Your sentence is:  This can be the test sentence
List string is :  ['This', 'can', 'be', 'the', 'test', 'sentence']
EVEN Length Words are:
This
be
test
sentence
```

```
1.  # Display Even length words in a Sentence
2.
3.  # Get String from User
4.  str = input("Please write a Sentence: ")
5.
6.  # Move the sentence into a List
7.  listx = list(str.split(' '))
8.
9.  print ("Your sentence is: ", str);
10.
11. # Converted list of words
12. print ("List string is : ", listx);
13.
14. # Check word by word get its length and print only when the word is EVEN
15. print ("EVEN Length Words are:");
16. for word in listx:
17.     if(len(word)%2==0):
18.         print (word);
```

Use Case 16

Use Case 17

Use Case 18

Use Case 19

Use Case 20

Use Case 21

Use Case 22

Use Case 23

Use Case 24

Use Case 25

Use Case 26

Use Case 27

Use Case 28

Use Case 29

Use Case 30