In [ ]:
```
#Today Agenda:
                    1. Continue to for loop
                    2. While Loop
                    3. Functions
                    4. Problem Solving.
```

In [ ]:
```
# Range() with for loop:
1. range(n) - generates the numbers from 0 to n-1
2. range(start,stop) - generates the numbers from start and stop-1
3. range(start,stop,step-size)- genearte the numbers with step-size
```

In [2]:
```
list(range(1,10))
```

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

In [9]:
```
print(list(range(0,100,2)),end=" ")
```

```
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 4
0, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 74, 76, 7
8, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98]
```

In [11]:
```
#For loop with else block:
for i in range(100):
    print(i,end=" ")
else:
    print("\n All items are iterated")
```

```
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 27 28
29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54
55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
 All items are iterated
```

In [12]:
```
#Nested for loop:
#if one loop is used in another for loop is called nested for loop.
for num1 in range(0,5):
    for num2 in range(10,20):
        print(num1,",",num2)
```

```
0 , 10
0 , 11
0 , 12
0 , 13
0 , 14
0 , 15
0 , 16
0 , 17
0 , 18
0 , 19
1 , 10
1 , 11
1 , 12
1 , 13
1 , 14
1 , 15
1 , 16
1 , 17
1 , 18
1 , 19
2 , 10
2 , 11
2 , 12
2 , 13
2 , 14
2 , 15
2 , 16
2 , 17
2 , 18
2 , 19
3 , 10
3 , 11
3 , 12
3 , 13
3 , 14
3 , 15
3 , 16
3 , 17
3 , 18
3 , 19
4 , 10
4 , 11
4 , 12
4 , 13
4 , 14
4 , 15
4 , 16
4 , 17
4 , 18
4 , 19
```

In [ ]:
```
#Break and Continue and Pass statements in for loop:
#Break:
its used to exit the for loop prematurely.
its used to break the for loop when a we met specific condition.
```

In [18]:
```
#Example:
li = [1,2,3,4,5,6]   #array type of data structure
n = 3
found = False
for num in li:
    if n == num:
        found = True
        break
else:
    print("for loop is terminated")
print(f'List Contains {n}:{found}')
```

```
List Contains 3:True
```

In [ ]:
```
#Continue:
we can use continue statements inside a for loop to skip the execution of the
for loop
body for a specific condition.
```

In [25]:
```
#example-continue:
nums = [1,2,-3,4,-5,6,-8]
sum_p = 0
for num in nums:
    if num < 0:          #skiping the negative numbers
        continue
    sum_p  = sum_p+num
print(f'sum of postive numbers:{sum_p}')
```

```
sum of postive numbers:13
```

In [ ]:
```
#pass statement:
Its like a null statement
The interpreter ignores the comment but pass is not ignored.
Its like a no operation(NOP).
```

In [32]:
```
li = [20,30,70,50,70,100]
for val in li:
     pass
```

In [ ]:
```
#while Loop:
Its used to iterate over a block of code repeatedly until a given condition re
turns False.
Note:
     difference b/w for loop and while loop:
            when we know the number of iterations we need to run the loop- use
for loop
            otherwise - please go with while loop.
```

In [ ]:
```
#Synatx:
while condition:
    #block of statements
```

In [1]:
```
#Example: to print the numbers from 1 to 10
num =100
while num >10:
    print(num,end= " ")
    num = num-2
```

```
100 98 96 94 92 90 88 86 84 82 80 78 76 74 72 70 68 66 64 62 60 58 56 54 52 5
0 48 46 44 42 40 38 36 34 32 30 28 26 24 22 20 18 16 14 12
```

In [ ]:
```

```

In [2]:
```
#Nested while loop:
i = 1
j = 5
while i <= 4:
    while j <= 8:
        print(i, ",",j)
        j = j+1
        i = i+1
```

```
1 , 5
2 , 6
3 , 7
4 , 8
```

In [35]:
```
#While loop else block:
num = 10
while num > 6:
    print(num)
    num = num-1
```

```
10
9
8
7
```

In [11]:
```
#Break with While Loop:
for num in [11,6,8,9,88,90,123]:
    print(num)
    while num==6:
        print("The number 6 is found")
        print("Terminating the loop")
        break
```

```
11
6
The number 6 is found
Terminating the loop
8
9
88
90
123
```

In [27]:
```
#Print the Values with index:
books = ["C","C++","java","python"]
for index in range(len(books)):
    print('Book(%d):' % index,books[index])
```

```
Book(0): C
Book(1): C++
Book(2): java
Book(3): python
```

In [ ]:
```
#Functions:
Function is a group of related statements that perfomrs a specific task.
why use functions?
 1. To avoid code repition
 2. To make our Complex program into smaller sub programs for easy degug.
 3. write once and Use any times. (code resuabilty)
```

In [ ]:
```
#How to create functions:
def funtionname(paraementers): # formal parameters
    """doc string"""
    #block of statements
    return
functionname(paramenters) #function calling with calling parameters(actual parameters)
```

In [22]:
```
import keyword
print((keyword.kwlist),end=" ")
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break',
'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'fo
r', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'no
t', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

In [33]:
```
#Example for Function:
def displayname(name):
    """This is a Function to Display Your Name"""   #Doc string
    print("Hello," + name +" Good Evening to All")
name = input("enter your name")  #dynamic parameter
displayname(name)
```

```
enter your namesurya
Hello,surya Good Evening to All
```

In [ ]:
```
#Day- 5 (Today Tasks)

1. find the given number is palindrome or not.
2. check the given number is prime or not.
3. Print the given year is leap or not.
4. print the leap years in given range of years.
5. print the math table as up to given number
6. To check the given number is positive or not.
7. Print the swaping of given actual numbers
8. Program to do the basic calculator operations.
```