In [ ]:
```
1.
->def keyword is used to create function
```

In [27]:
```
l=[]
for i in range (1,26):
    l.append(i)
    i=i+1

l2=[]
def return_odd(l):
    for i in range (len(l)):
        if l[i]%2!=0:
           l2.append(l[i])
           i=i+1
        else:
            pass
    return l2

print(return_odd(l))
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25]
```

In [ ]:
```
2.
->*args and **kwargs are used when you are unsure about the number of arguments to
```

In [3]:
```
def args_funct(*args):
    for i in args :
        print(i)

args_funct("hello","welcome","to","pwskills ")
```

```
hello
welcome
to
pwskills
```

In [4]:
```
def kwargs_funct(**kwargs):
    for key, value in kwargs.items():
        print("%s==%s"%(key,value))

kwargs_funct(first="name",mid="id",last="mob_no")
```

```
first==name
mid==id
last==mob_no
```

In [ ]:
```
3.
->an iterator is an object that contains a countable number of values .eg lists,tup
 the iterator object is initialized using the iter() method
 it uses the next() method for iteration
```

In [9]:
```
list=[2,4,6,8,10,12,14,16,18,20]
l_iterator=iter(list)
i=0
try:
    while i<5:
        element=next(l_iterator)
        print(element)
        i=i+1
except stopiteration:
    pass
```

```
2
4
6
8
10
```

In [ ]:
```
4.what is generator function in python ?why yield keyword is used ?give an example
->generator is function that returns an iterator that produces a sequence of values
  yield keyword is used to create a generator function
```

In [10]:
```
#eg
def generator_funct():
    yield 1
    yield 2
    yield 3

for i in generator_funct():
    print(i)
```

```
1
2
3
```

In [1]:
```
5.
from math import sqrt

def is_prime(n):
    if (n<=1):
        return False
    if (n==2):
        return True
    if (n%2==0):
        return False

    i=3
    while i<=sqrt(n):
        if n%i==0:
            return False
        i=i+2

    return True


def prime_generator(limit=1000):
    n=1
    while True:
        if n<limit:
            if is_prime(n):
                yield n
            n+=1
        else:
            break
generator=prime_generator(1000)

for i in range (20):
    print(next(generator))
```

```
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
```

In [1]:
```
6.
a,b=0,1

count=10
print("fibonacci series: ",a,b,end=" ")
for i in range (2, count):
    c=a+b

    a=b
    b=c
    print(c,end=" ")
print()
```

```
fibonacci series:  0 1 1 2 3 5 8 13 21 34
```

In [1]:
```
7.
string="pwskiils"
list_com=[i for i in string ]
print(list_com)
```

```
['p', 'w', 's', 'k', 'i', 'i', 'l', 's']
```

In [5]:
```
8.
num=int(input("enter your no: "))
temp=num
reversed_num=0

while num!=0:
    digit=num%10
    reversed_num=reversed_num*10+digit
    num//=10

if reversed_num==temp:
    print("this is an pallindrome no.")
else:
    print("this is not an pallindrome no.")
```

```
this is an pallindrome no.
```

In [6]:
```
9.
list_com=[i for i in range (1,101) if i%2!=0]
print(list_com)
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43,
45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75, 77, 79, 81, 83, 85,
87, 89, 91, 93, 95, 97, 99]
```

In [ ]: