Untitled

April 7, 2023

1 question 1

1.0.1 the function is created by the def

[ ]:

```
```

[23]:

```
def test2(a,b):
    n=[]
    for i in range(a,b) :
        if i %2 !=0:
            n.append(i)
    return n
```

[24]:

```
test2(1,25)
```

[24]:

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

[ ]:

```
```

1.1 question 2

*args is an argument, given to have multiple number of inputs**kwargs is an key-word argument mentioned for dictionaries

[26]:

```
def test3(*args):
    return args
```

[27]:

```
test3(2)
```

[27]:

```
(2,)
```

[28]:

```
test3([1,2,44,55,4],"punith",True)
```

[28]:

```
([1, 2, 44, 55, 4], 'punith', True)
```

[ ]:

```
```

[29]:
```
def test4(**kwargs):
    return kwargs
```

[34]:
```
test4(a=9,b=70,c=65)
```

[34]:
```
{'a': 9, 'b': 70, 'c': 65}
```

[38]:
```
def test5(*args,**kwargs):
    return args,kwargs
```

[39]:
```
test5([1,2,3,4,5,66],(1,7,8,9,6),a=87,b="punith",c="datascience",d=True)
```

[39]:
```
(([1, 2, 3, 4, 5, 66], (1, 7, 8, 9, 6)),
 {'a': 87, 'b': 'punith', 'c': 'datascience', 'd': True})
```

[ ]:
```

```

1.2 3 question

[41]:
```
l=[2, 4, 6, 8, 10, 12, 14, 16,18, 20]
```

[48]:
```
def test6(a):
    n=[]
    for x in a:
        if x<=12:
            n.append(x)
    return n
```

[49]:
```
test6(l)
```

[49]:
```
[2, 4, 6, 8, 10, 12]
```

[ ]:
```

```

1.2.1 question 4

[ ]:
```
A Python generator function allows you to declare a function that behaves like an iterator, providing a faster and easier way to create iterators.
```

[ ]:
```
yeild statement is  similar to a return statement and  used for returning values or objects in Python.
```

[50]:
```
range(1,10)
```

[50]:
```
range(1, 10)
```

[53]:

```
for i in range(1,10):
    print(i)
```

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

1.3 fib numbers:

0,1,2,3,5,8,13,21,34……….n

[110]:

```
def fib1(x):
    a,b=0,1
    for i in range(x):
        yield a
        a,b=b,a+b

```

[116]:

```
fib1(100)
```

[116]:

```
<generator object fib1 at 0x7f18302e4120>
```

[117]:

```
for i in fib1(10):
    print(i)
```

```
0
1
1
2
3
5
8
13
21
34
```

[ ]:

```

```

[ ]:

```

```

[ ]:

```

```

[ ]:

[ ]:

[ ]:

[ ]: