Untitled

April 6, 2023

[2]:
```
print("this is my programme")
```

```
this is my programme
```

0.0.1 How to create our own functions?

0.0.2 Step 1 ; writing def(definition)

[27]:
```
def test():
    return
```

[28]:
```
## above is the formt of writing our own functions
```

[29]:
```
test() + "punith"
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[29], line 1
----> 1 test() + "punith"

TypeError: unsupported operand type(s) for +: 'NoneType' and 'str'
```

[9]:
```
## the concatenation functions are not allowed to perform
```

[10]:
```
def test2():
    return "this is my functions"
```

[11]:
```
test2()
```

[11]:
```
'this is my functions'
```

[14]:
```
test2() + " Punith "
```

[14]:
```
'this is my functions Punith '
```

[50]:
```
### aways use the return functions instead of print for writing the ur own␣
↪functions
```

```
## because it has to return the particular functions
```

[30]:
```
def test3():
    return 1,2,3,4,"punith",True,340.25
```

[31]:
```
test3()
```

[31]: (1, 2, 3, 4, 'punith', True, 340.25)

[41]:
```
## the outcome is in the form of tuples
```

[35]:
```
a= 1,2,3,4,5,66
```

[36]:
```
a
```

[36]: (1, 2, 3, 4, 5, 66)

[37]:
```
a,b,c,d,e = 1,2,5,4,5
```

[38]:
```
a
```

[38]: 1

[39]:
```
b
```

[39]: 2

[40]:
```
c
```

[40]: 5

[42]:
```
test3()
```

[42]: (1, 2, 3, 4, 'punith', True, 340.25)

[43]:
```
test3()[0]
```

[43]: 1

[44]:
```
test3()[5]
```

[44]: True

[45]:
```
### the process of extracting the element
```

[48]:
```
test3().index("punith")
```

[48]: 4

[49]:
```
## even indexing can be done
```

[54]:
```
a,b,c,d,e,f,g = test3()
```

[55]:
```
a
```

[55]: 1

[56]:
```
e
```

[56]: 'punith'

[57]:
```
f
```

[57]: True

[58]:
```
g
```

[58]: 340.25

[73]:
```
## example 1
```

[59]:
```
def test4():
    a =3*4+5
    return a
```

[60]:
```
test4()
```

[60]: 17

[70]:
```
## example 2
```

[65]:
```
def test5(a,b):
    c =a+b
    return c


```

[66]:
```
test5("punith","Kumar")
```

[66]: 'punithKumar'

[67]:
```
##  test5 is acting as a function
```

[68]:
```
test5(4,5)
```

[68]:
```
9
```

[71]:
```
test5([1,2,3],[4,5,6])
```

[71]:
```
[1, 2, 3, 4, 5, 6]
```

[75]:
```
test5("kumar","punith")
```

[75]:
```
'kumarpunith'
```

[76]:
```
## or
```

[78]:
```
test5(b="punith",a="kumar")
```

[78]:
```
'kumarpunith'
```

[79]:
```
## example 3
```

[80]:
```
l=[1,2,3,4,5,6,"punith","pwskills",[44,5,5,6,8]]
```

[93]:
```
## create a function to filter only numeric value?
```

[88]:
```
def test6(a):
    n=[]
    for x in a:
        if type(x)==int:
            n.append(x)
    return n
```

[89]:
```
test6(l)
```

[89]:
```
[1, 2, 3, 4, 5, 6]
```

[92]:
```
## n is my new list
## append means add that into my new list
```

[ ]:
```

```

[95]:
```
## extract all int even from the nested list?
```

[96]:
```
l
```

[96]:
```
[1, 2, 3, 4, 5, 6, 'punith', 'pwskills', [44, 5, 5, 6, 8]]
```

[110]:
```
def test7(a):
    n=[]
    for x in a:
```

```
        if type(x)== list:
            for j in x:
                if type(j)==int:
                    n.append(j)
            return n
        else:
            if type(x)==int:
                n.append(x)
    return n
```

[111]:
```
test7(l)
```

[111]:
```
[1, 2, 3, 4, 5, 6, 44, 5, 5, 6, 8]
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: