Best practices for Python Hands-on coding under ION lite environment

Simple guidelines, to be followed to prevent indentation issues:

- Maintain the **uniform indentation method, ie., either (space(s)/tab) for indentation level** throughout your program. Don’t use the combination of tabs and spaces to represent the indentation in a program,
- Always prefer space(s) for indentation, as different applications (Notepad,word,Editor) would have different tab configurations in terms of **number of spaces** for a tab,
- Use the **single space** to differentiate **one level of indentation** with **another level. If you use multiple spaces then** there is a chance to miss or add extra space while typing,
- Avoiding indentation errors, comes with more practice. Practice as many problem statements as you.

Lets understand with the Example, Where in you would require to implement a **Class with functions and some code in each of the functions :**

Lets assume that the class declaration (**class Account:)** **started** at **first column in the first line** of the coding editor , not only ION lite any where **.**

Then the **Functions declaration/definitions** of the **class Account ,** should start in the new line **with a**n extra **space** after the place, where the **class Account:** started **.** ie with one extra space from the place where the class started.

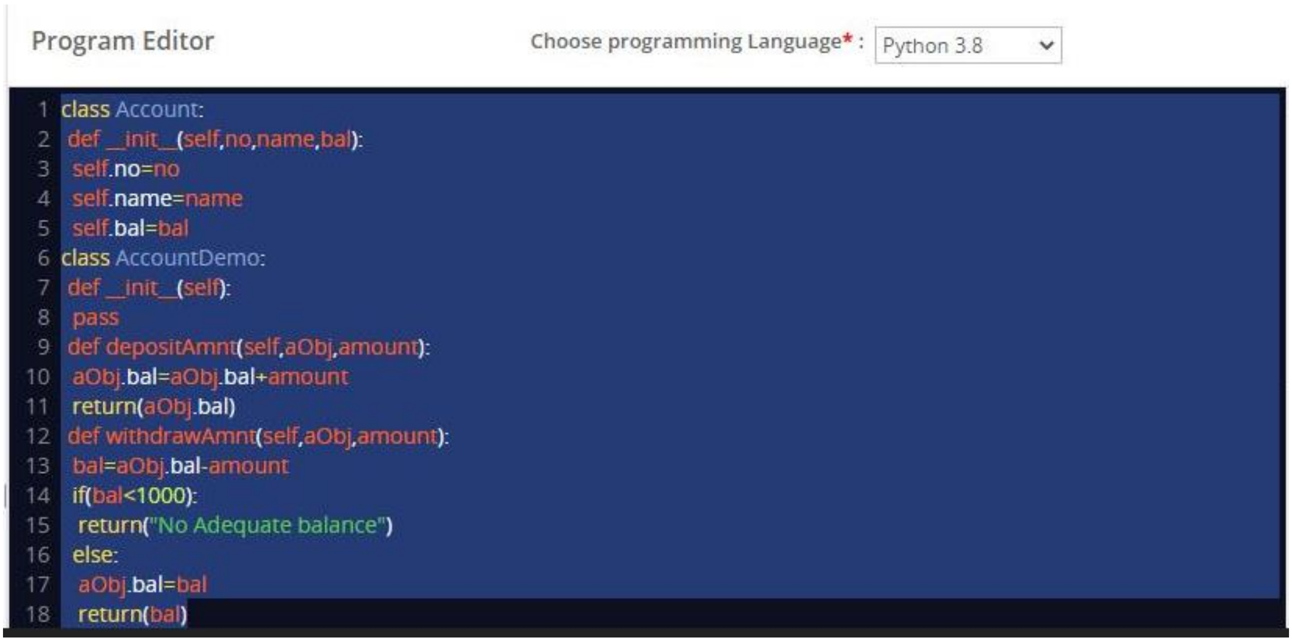
Accordingly the **Logic inside any function of the respective class** should start in new lines **with a space after the place** ,where the respective function declaration/definition started.
ie two extra spaces from the place, where the respective class(to which this function belongs to)

Snippet for the above guidelines for Example: Refer the comment for each line of code
****

```
class Account: #This class declaration started as the first line at the first column in the line
 def __init__(self,no,name,bal):#This line starts with an extra space from the place, where class starts
  self.no=no #This line starts with an extra space from the place, where its function(__init__) starts
  self.name=name#This starts with an extra space from the place,where its function(__init__) starts
  self.bal=bal #This line starts with an extra space from the place,where its function(__init__) starts

#Similarly refer teh below code with indentation
class AccountDemo:
 def __init__(self):
  pass
 def depositAmnt(self,aobj,amount):
  aobj.bal=aobj.bal+amount
  return(aobj.bal)
```

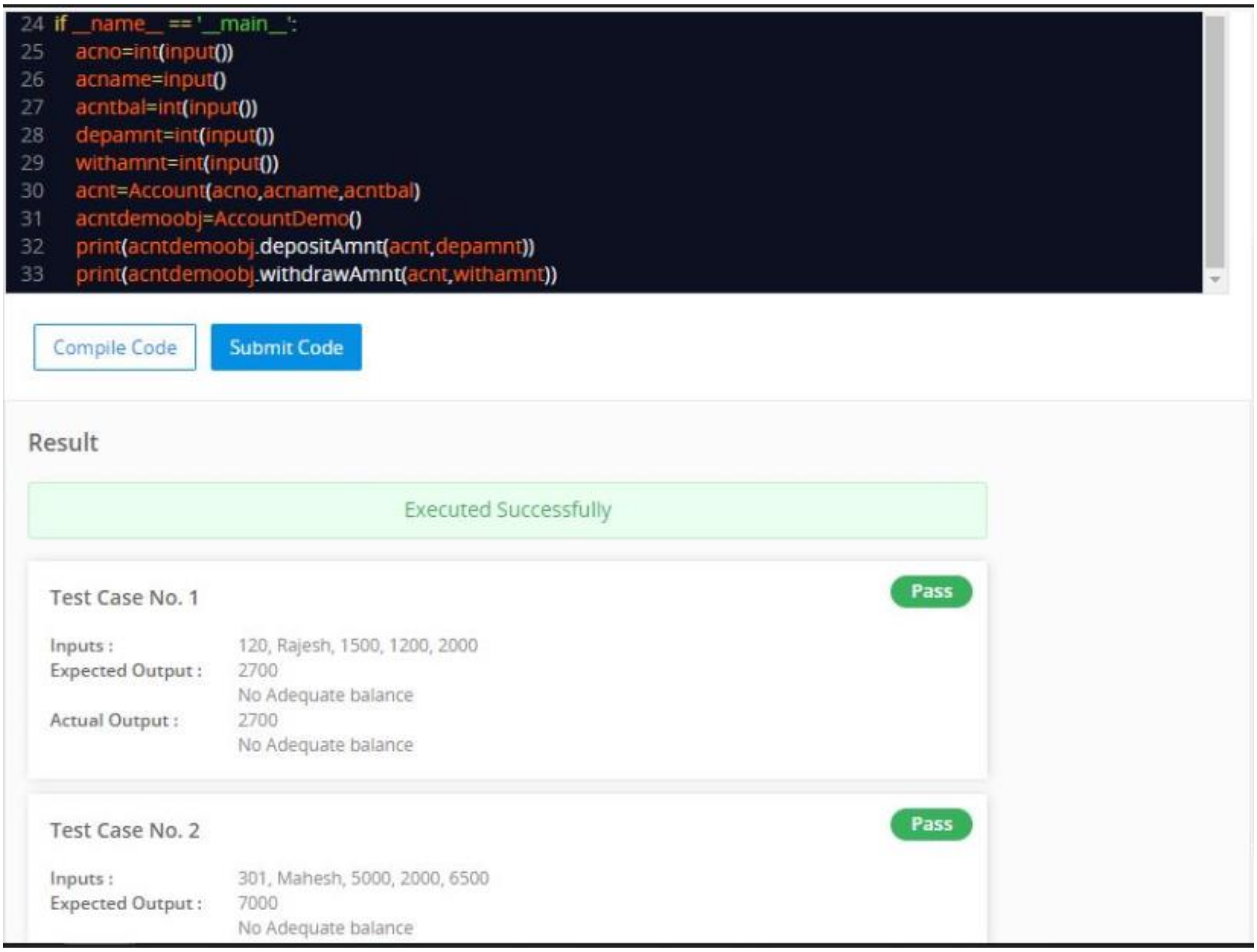
Note:
You can also follow with multiple spaces (two or three) to identify the one level of indentation from other, But needs to be careful while applying and the indentation method should be uniform to differentiate one level to another level of indentation across the program as mentioned above . Then you never get indentation issues.
Please find the ION-lite editor snap shots for the trainees to refer for more clarity for the Account Problem:



Program Editor

Choose programming Language* : Python 3.8

```
class Account:
 def __init__(self,no,name,bal):
  self.no=no
  self.name=name
  self.bal=bal
class AccountDemo:
 def __init__(self):
  pass
 def depositAmnt(self,aObj,amount):
  aObj.bal=aObj.bal+amount
  return(aObj.bal)
 def withdrawAmnt(self,aObj,amount):
  bal=aObj.bal-amount
  if(bal<1000):
   return("No Adequate balance")
  else:
   aObj.bal=bal
   return(bal)
```



```
if __name__ == '__main__':
    acno=int(input())
    acname=input()
    acntbal=int(input())
    depamnt=int(input())
    withamnt=int(input())
    acnt=Account(acno,acname,acntbal)
    acntdemoobj=AccountDemo()
    print(acntdemoobj.depositAmnt(acnt,depamnt))
    print(acntdemoobj.withdrawAmnt(acnt,withamnt))
```

Result

Executed Successfully

Test Case No. 1 Pass

Inputs : 120, Rajesh, 1500, 1200, 2000

Expected Output : 2700
No Adequate balance

Actual Output : 2700
No Adequate balance

Test Case No. 2 Pass

Inputs : 301, Mahesh, 5000, 2000, 6500

Expected Output : 7000
No Adequate balance

All the best ..