Lesson 10: Question 3

Check the following Student class code:

```
class Student:
    def __init__(self,fName,lName,reg):
        self.fName=fName
        self.lName=lName
        self.reg=reg
        self.fullName=f'{self.fName} {self.lName}'
    @property
    def fullName(self):
        return f'{self.fName} {self.lName}'
    @fullName.setter
    def fullName(self,newname):
        f,l=newname.split(' ')
        self.fName=f
        self.lName=l
        self._fullName=newname
    @property
    def fName(self):
        return self._fName
    @fName.setter
    def fName(self,newname):
        if(Student.validName(newname)):
            self._fName=newname
        else:
            raise ValueError('Name must contain alphabets only and at
least two of those!')

    @property
    def lName(self):
        return self._lName
    @lName.setter
    def lName(self,newname):
        if(Student.validName(newname)):
            self._lName=newname
        else:
            raise ValueError('Name must contain alphabets only and at
least two of those!')

    @property
    def reg(self):
        return self._reg
    @reg.setter
    def reg(self,newreg):
```

```
        if(isinstance(newreg,str) and str(newreg).startswith('MCT-
UET-')):
            self._reg=newreg
        else:
            raise ValueError('Reg. No. must start with MCT-UET-')

    @staticmethod
    def validName(newname):
        if(isinstance(newname,str) and len(newname)>=2 and newname.is
alpha()):
            return True
        else:
            return False
```

If we run the following Main Program:

```
std1=Student('Anwar','Ali','MCT-UET-01')
std2=Student('Akbar','Khan','MCT-UET-02')
print(std1.__dict__)
```

We will see fullName in Dictionary as:

```
{'_fName': 'Anwar', '_lName': 'Ali', '_reg': 'MCT-UET-01', '_fullName':
'Anwar Ali'}
```

But if we update just fName, will fullName get updated? Check with this code:

```
std1=Student('Anwar','Ali','MCT-UET-01')
std2=Student('Akbar','Khan','MCT-UET-02')

std1.fName='Ahmad'
print(std1.__dict__)
```

fName is updated for std1, but the dictionary shows the previous value as:

```
{'_fName': 'Ahmad', '_lName': 'Ali', '_reg': 'MCT-UET-01', '_fullName':
'Anwar Ali'}
```

So how can we make it get updated?

Firstly, why fullName is not updated after we updated the fName? Because when we use std1.fName='Ahmad', this will call the setter method for fName. Inside that setter method, just fName is updated and nothing happens to fullName and hence that stays same.

The first approach I thought was to update fullName in setter of fName and lName as:

```
@fName.setter
    def fName(self,newname):
        if(Student.validName(newname)):
            self._fName=newname
            self._fullName=f'{self.fName} {self.lName}'
        else:
            raise ValueError('Name must contain alphabets only and at
least two of those!')
```

When I did that in today's live session, it generated error. Let's see why it generated error. With above setter method, consider the init method of the class:

```
    def __init__(self,fName,lName,reg):
        self.fName=fName
        self.lName=lName
        self.reg=reg
        self.fullName=f'{self.fName} {self.lName}'
```

When an object is created, init method is called. And check carefully the first statement of the method i.e. self.fName=fName. This is going to call fName setter method and that will execute the statement highlight in arrow and that involves self.lName and lName is not yet defined.

So, we cannot do it like that.

A couple of students indicated a solution which keeps the fName setter unchanged but in main program they are doing it like:

```
std1=Student('Anwar','Ali','MCT-UET-01')
std2=Student('Akbar','Khan','MCT-UET-02')

std1.fName='Ahmad'
std1.fullName
print(std1.__dict__)
```

See the second last line which is needed just to get fullName updated. That line is doing nothing but will updated the fullName and hence dictionary will have updated fullName.

This is NOT proper solution as we will have to do it in Main Program and it is not automatic update.

Huzaifa Imtiaz did it correctly which you can find from YouTube comments and you will see that it's very complex.

So, here is my suggestion:

Dictionary is meant for the developers and not the end users. So, we should not worry so much about having the updated value in Dictionary. If a developer is debugging a program using our created class, he will get the information of object from dictionary and if interested he will access that directly.