Assignment_04_20101197_Abir_Ahammed_Bhuiyan

October 3, 2022

[1]:
```python
#oop_method_1

class Calculator:
    def __init__(self):
        print("Let's Calculate!")
        self.value1 = int(input("Value 1: "))
        self.operator = input("Operator: ")
        self.value2 = int(input("Value 2: "))
        if(self.operator == '+'):
            result = self.add(self.value1, self.value2)
        elif(self.operator == '-'):
            result = self.subtract(self.value1, self.value2)
        elif(self.operator == 'x' or self.operator == '*'):
            result = self.multiply(self.value1, self.value2)
        elif(self.operator == '/'):
            result = self.divide(self.value1, self.value2)
        else:
            result = "Something went wrong"
        print("Result:", result)

    def add(self, value1, value2):
        return value1+value2
    def subtract(self, value1, value2):
        return value1-value2
    def multiply(self, value1, value2):
        return value1*value2
    def divide(self, value1, value2):
        return value1/value2

obj1 = Calculator()
```

```
Let's Calculate!

Value 1:  1
Operator:  +
Value 2:  2

Result: 3
```

[2]:
```
#oop_method_2

class Customer:
    def __init__(self, name):
        self.name = name

    def greet(self, name=None):
        if(name!=None):
            print("Hello "+name+"!")
        else:
            print("Hello!")

    def purchase(self, *args):
        print(self.name+", you purchased "+str(len(args))+" item(s):")
        for arg in args:
            print(arg)



customer_1 = Customer("Sam")
customer_1.greet()
customer_1.purchase("chips", "chocolate", "orange juice")
print("---------------------")
customer_2 = Customer("David")
customer_2.greet("David")
customer_2.purchase("orange juice")
```

```
Hello!
Sam, you purchased 3 item(s):
chips
chocolate
orange juice
---------------------
Hello David!
David, you purchased 1 item(s):
orange juice
```

[3]:
```
#oop_method_3

class Panda:
    def __init__(self, name, gender, age):
        self.name = name
        self.gender = gender
        self.age = age

    def sleep(self, hour=None):
        if(hour!=None):
            if(hour>=9 and hour<=11):
                return self.name+" sleeps "+str(hour)+" hours daily and should␣
 ↪have Broccoli Chicken"
            elif(hour>=6 and hour<=8):
                return self.name+" sleeps "+str(hour)+" hours daily and should␣
 ↪have Eggplant & Tofu"
            elif(hour>=3 and hour<=5):
                return self.name+" sleeps "+str(hour)+" hours daily and should␣
 ↪have Mixed Veggies"
        else:
            return self.name+"'s duration is unknown thus should have only␣
 ↪bamboo leaves"



panda1 = Panda("Kunfu", "Male", 5)
panda2 = Panda("Pan Pan", "Female", 3)
panda3 = Panda("Ming Ming", "Female", 8)


print("{} is a {} Panda Bear who is {} years old".format(panda1.name, panda1.
 ↪gender, panda1.age))

print("{} is a {} Panda Bear who is {} years old".format(panda2.name, panda2.
 ↪gender, panda2.age))


print("{} is a {} Panda Bear who is {} years old".format(panda3.name, panda3.
 ↪gender, panda3.age))


print("========================")

print(panda2.sleep(10))
print(panda1.sleep(4))
print(panda3.sleep())
```

```
Kunfu is a Male Panda Bear who is 5 years old
Pan Pan is a Female Panda Bear who is 3 years old
Ming Ming is a Female Panda Bear who is 8 years old
========================
Pan Pan sleeps 10 hours daily and should have Broccoli Chicken
Kunfu sleeps 4 hours daily and should have Mixed Veggies
Ming Ming's duration is unknown thus should have only bamboo leaves
```

[4]:
```python
#oop_method_4

class Cat:
    def __init__(self, color=None, verb=None):
        if(color == None):
            self.color = "White"
        else:
            self.color = color

        if(verb==None):
            self.verb = "sitting"
        else:
            self.verb = verb

    def printCat(self):
        print(self.color, "cat is", self.verb)

    def changeColor(self, color):
        self.color = color





c1 = Cat()
c2 = Cat("Black")
c3 = Cat("Brown", "jumping")
c4 = Cat("Red", "purring")

c1.printCat()
c2.printCat()
c3.printCat()
c4.printCat()

c1.changeColor("Blue")
c3.changeColor("Purple")
c1.printCat()
c3.printCat()
```

```
White cat is sitting
Black cat is sitting
Brown cat is jumping
Red cat is purring
Blue cat is sitting
Purple cat is jumping
```

[5]:
```
#oop_method_5

class Student:
    def __init__(self, name=None):
        if(name == None):
            self.name = "default student"
        else:
            self.name = name

        self.avg = 0

    def quizcalc(self, *marks):
        sum = 0
        for mark in marks:
            sum += int(mark)
        self.avg = sum/3

    def printdetail(self):
        print("Hello", self.name)
        print("Your average quiz score is", str(self.avg))


s1 = Student()
s1.quizcalc(10)
print('-----------------------')
s1.printdetail()
s2 = Student('Harry')
s2.quizcalc(10, 8)
print('-----------------------')
s2.printdetail()
s3 = Student('Hermione')
s3.quizcalc(10, 9, 10)
print('-----------------------')
s3.printdetail()
```

```
-----------------------
Hello default student
Your average quiz score is 3.3333333333333335
-----------------------
Hello Harry
Your average quiz score is 6.0
-----------------------
Hello Hermione
Your average quiz score is 9.666666666666666
```

[6]:
```
#oop_method_6
```

```python
class Vehicle:
    def __init__(self):
        self.x = 0
        self.y = 0

    def moveUp(self):
        self.y+=1
    def moveDown(self):
        self.y-=1
    def moveRight(self):
        self.x+=1
    def moveLeft(self):
        self.x-=1
    def print_position(self):
        print(f"({self.x}, {self.y})")

car = Vehicle()
car.print_position()
car.moveUp()
car.print_position()
car.moveLeft()
car.print_position()
car.moveDown()
car.print_position()
car.moveRight()
```

```
(0, 0)
(0, 1)
(-1, 1)
(-1, 0)
```

[7]:
```python
#oop_method_7

class Programmer:
    def __init__(self, name, language, xp):
        self.name = name
        self.language = language
        self.xp = xp
        print("Horray! A new programmer is born")

    def addExp(self, new_xp):
        self.xp += new_xp
        print("Updating experience of", self.name)

    def printDetails(self):
        print("Name:", self.name)
        print("Language:", self.language)
        print("Experience:", self.xp, "years.")


p1 = Programmer("Ethen Hunt", "Java", 10)
p1.printDetails()
print('----------------------')
p2 = Programmer("James Bond", "C++", 7)
p2.printDetails()
print('----------------------')
p3 = Programmer("Jon Snow", "Python", 4)
p3.printDetails()
p3.addExp(5)
p3.printDetails()
```

```
Horray! A new programmer is born
Name: Ethen Hunt
Language: Java
Experience: 10 years.
----------------------
Horray! A new programmer is born
Name: James Bond
Language: C++
Experience: 7 years.
----------------------
Horray! A new programmer is born
Name: Jon Snow
Language: Python
Experience: 4 years.
Updating experience of Jon Snow
Name: Jon Snow
Language: Python
Experience: 9 years.
```

[8]:
```
#oop_method_8

class Student:
    def __init__(self, name, ID, dept="CSE"):
        self.name = name
        self.ID = ID
        self.dept = dept

    def dailyEffort(self, deff):
        self.deff = deff


    def printDetails(self):
        print("Name:", self.name)
        print("ID:", self.ID)
        print("Department:", self.dept)
        print("Daily Effort:", self.deff, "hour(s)")
        if(self.deff<=2):
            print("Suggestion: Should give more effort!")
        elif(self.deff<=4):
            print("Suggestion: Keep up the good work!")
        else:
            print("Suggestion: Excellent! Now motivate others.")



harry = Student('Harry Potter', 123)
harry.dailyEffort(3)
harry.printDetails()
print('========================')
john = Student("John Wick", 456, "BBA")
john.dailyEffort(2)
john.printDetails()
print('========================')
naruto = Student("Naruto Uzumaki", 777, "Ninja")
naruto.dailyEffort(6)
naruto.printDetails()
```

```
Name: Harry Potter
ID: 123
Department: CSE
Daily Effort: 3 hour(s)
Suggestion: Keep up the good work!
========================
Name: John Wick
ID: 456
Department: BBA
Daily Effort: 2 hour(s)
Suggestion: Should give more effort!
========================
Name: Naruto Uzumaki
ID: 777
Department: Ninja
Daily Effort: 6 hour(s)
Suggestion: Excellent! Now motivate others.
```

[9]:
```
#oop_method_9

class Patient:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def add_Symptom(self, *args):
        self.symp = ""
        for arg in args:
            self.symp+=arg+", "
        self.symp = self.symp[:-2]

    def printPatientDetail(self):
        print("Name:", self.name)
        print("Age:", self.age)
        print("Symptoms:", self.symp)



p1 = Patient("Thomas", 23)
p1.add_Symptom("Headache")
p2 = Patient("Carol", 20)
p2.add_Symptom("Vomiting", "Coughing")
p3 = Patient("Mike", 25)
p3.add_Symptom("Fever", "Headache", "Coughing")
print("=========================")
p1.printPatientDetail()
print("=========================")
p2.printPatientDetail()
print("=========================")
p3.printPatientDetail()
print("=========================")
```

```
=========================
Name: Thomas
Age: 23
Symptoms: Headache
=========================
Name: Carol
Age: 20
Symptoms: Vomiting, Coughing
=========================
Name: Mike
Age: 25
Symptoms: Fever, Headache, Coughing
=========================
```

[10]:
```
#oop_method_10

class Avengers:
    def __init__(self, name, partner):
```

```python
        self.name = name
        self.partner = partner

    def super_powers(self, *args):
        self.suppow = ""
        for arg in args:
            self.suppow += arg + ", "
        self.suppow = self.suppow[:-2]

    def printAvengersDetail(self):
        print("Name:", self.name)
        print("Partner:", self.partner)
        print("Super powers:", self.suppow)


a1 = Avengers('Captain America', 'Bucky Barnes')
a1.super_powers('Stamina', 'Slowed ageing')
a2 = Avengers('Doctor Strange', 'Ancient One')
a2.super_powers('Mastery of magic')
a3 = Avengers('Iron Man', 'War Machine')
a3.super_powers('Genius level intellect', 'Scientist ')
print("=========================")
a1.printAvengersDetail()
print("=========================")
a2.printAvengersDetail()
print("=========================")
a3.printAvengersDetail()
print("=========================")
```

```
=========================
Name: Captain America
Partner: Bucky Barnes
Super powers: Stamina, Slowed ageing
=========================
Name: Doctor Strange
Partner: Ancient One
Super powers: Mastery of magic
=========================
Name: Iron Man
Partner: War Machine
Super powers: Genius level intellect, Scientist
=========================
```

[11]:
```python
#oop_method_11

class Shinobi:
    def __init__(self, name=None, rank=None):
        self.name = name
        self.rank = rank
        self.salary = 0
        self.mission = 0

    def calSalary(self, mission):
        self.mission = mission
        if(self.rank == 'Genin'):
            self.salary = self.mission*50
        elif(self.rank == 'Chunin'):
            self.salary = self.mission*100
        else:
            self.salary = self.mission*500

    def printInfo(self):
        print("Name:", self.name)
        print("Rank:", self.rank)
        print("Number of mission:", self.mission)
        print("Salary:", self.salary)

    def changeRank(self, new_rank):
        self.rank = new_rank


naruto = Shinobi("Naruto", "Genin")
naruto.calSalary(5)
naruto.printInfo()
print('====================')
shikamaru = Shinobi('Shikamaru', "Genin")
shikamaru.printInfo()
shikamaru.changeRank("Chunin")
shikamaru.calSalary(10)
shikamaru.printInfo()
print('====================')
neiji = Shinobi("Neiji", "Jonin")
neiji.calSalary(5)
neiji.printInfo()
```

```
Name: Naruto
Rank: Genin
Number of mission: 5
Salary: 250
====================
Name: Shikamaru
Rank: Genin
Number of mission: 0
Salary: 0
```

```
Name: Shikamaru
Rank: Chunin
Number of mission: 10
Salary: 1000
====================
Name: Neiji
Rank: Jonin
Number of mission: 5
Salary: 2500
```

[12]:
```python
#oop_method_12

class ParcelKoro:
    def __init__(self, name="No name set", product_weight=0):
            self.name = name
            self.product_weight = product_weight

    def calculateFee(self, loc_name=None):
        if(self.product_weight == 0):
            self.total_fee = 0
        else:
            if(loc_name != None):
                self.total_fee = (self.product_weight*20) + 100
            else:
                self.total_fee = (self.product_weight*20) + 50

    def printDetails(self):
        print("Cutomer Name:", self.name)
        print("Product Weight:", self.product_weight)
        print("Total fee:", self.total_fee)



print("**********************")
p1 = ParcelKoro()
p1.calculateFee()
p1.printDetails()
print("**********************")
p2 = ParcelKoro('Bob The Builder')
p2.calculateFee()
p2.printDetails()
print("----------------------------")
p2.product_weight = 15
p2.calculateFee()
p2.printDetails()
print("**********************")
p3 = ParcelKoro('Dora The Explorer', 10)
```

```
p3.calculateFee('Dhanmondi')
p3.printDetails()
```

```
**********************
Cutomer Name: No name set
Product Weight: 0
Total fee: 0
**********************
Cutomer Name: Bob The Builder
Product Weight: 0
Total fee: 0
----------------------------
Cutomer Name: Bob The Builder
Product Weight: 15
Total fee: 350
**********************
Cutomer Name: Dora The Explorer
Product Weight: 10
Total fee: 300
```

[13]:
```
#oop_method_13

class Batsman:
    def __init__(self, *args):
        if(len(args) == 2):
            self.name = "New Batsman"
            self.runs_scored = args[0]
            self.balls_faced = args[1]
        else:
            self.name = args[0]
            self.runs_scored = args[1]
            self.balls_faced = args[2]

    def printCareerStatistics(self):
        print("Name:", self.name)
        print(f"Runs Scored: {self.runs_scored} ,Balls Faced: {self.
 ↪balls_faced}")

    def battingStrikeRate(self):
        return (self.runs_scored/self.balls_faced)*100

    def setName(self, name):
        self.name = name

b1 = Batsman(6101, 7380)
b1.printCareerStatistics()
print("============================")
```

```
b2 = Batsman("Liton Das", 678, 773)
b2.printCareerStatistics()
print("----------------------------")
print(b2.battingStrikeRate())
print("============================")
b1.setName("Shakib Al Hasan")
b1.printCareerStatistics()
print("----------------------------")
print(b1.battingStrikeRate())
```

```
Name: New Batsman
Runs Scored: 6101 ,Balls Faced: 7380
============================
Name: Liton Das
Runs Scored: 678 ,Balls Faced: 773
----------------------------
87.71021992238033
============================
Name: Shakib Al Hasan
Runs Scored: 6101 ,Balls Faced: 7380
----------------------------
82.66937669376694
```

[14]:
```
#oop_method_14

class EPL_Team:
    def __init__(self, team_name, team_song="No Slogan"):
        self.team_name = team_name
        self.team_song = team_song
        self.no_of_title = 0


    def increaseTitle(self):
        self.no_of_title += 1

    def changeSong(self, team_song):
        self.team_song = team_song

    def showClubInfo(self):
        return f"Name: {self.team_name}\nSong: {self.team_song}\nTotal no of␣
 ↪title: {self.no_of_title}"


manu = EPL_Team('Manchester United', 'Glory Glory Man United')
chelsea = EPL_Team('Chelsea')
print('==================')
print(manu.showClubInfo())
print('##################')
manu.increaseTitle()
print(manu.showClubInfo())
print('===================')
print(chelsea.showClubInfo())
chelsea.changeSong('Keep the blue flag flying high')
print(chelsea.showClubInfo())
```

```
===================
Name: Manchester United
Song: Glory Glory Man United
Total no of title: 0
##################
Name: Manchester United
Song: Glory Glory Man United
Total no of title: 1
===================
Name: Chelsea
Song: No Slogan
Total no of title: 0
Name: Chelsea
Song: Keep the blue flag flying high
Total no of title: 0
```

[15]:
```
#oop_method_15

class Account:
    def __init__(self, name="Default Account", balance=0.0):
        self.name = name
        self.balance = float(balance)

    def details(self):
        return f"Name: {self.name}\n{self.balance}"

    def withdraw(self, amnt):
        if((self.balance-amnt)<=(0.307*self.balance)):
            print("Sorry, Withdraw unsuccessful! The account balance after␣
↪deducting withdraw amount is equal to or less than minimum.")
        else:
            print("Withdraw successful! New Balance is:", self.balance-amnt)




a1 = Account()
print(a1.details())
print("---------------------")
```

```
a1.name = "Oliver"
a1.balance = 10000.0
print(a1.details())
print("--------------------")
a2 = Account("Liam")
print(a2.details())
print("--------------------")
a3 = Account("Noah", 400)
print(a3.details())
print("--------------------")
a1.withdraw(6930)
print("--------------------")
a2.withdraw(600)
print("--------------------")
a1.withdraw(6929)
```

```
Name: Default Account
0.0
--------------------
Name: Oliver
10000.0
--------------------
Name: Liam
0.0
--------------------
Name: Noah
400.0
--------------------
Sorry, Withdraw unsuccessful! The account balance after deducting withdraw
amount is equal to or less than minimum.
--------------------
Sorry, Withdraw unsuccessful! The account balance after deducting withdraw
amount is equal to or less than minimum.
--------------------
Withdraw successful! New Balance is: 3071.0
```

[16]:
```
#oop_method_16

class Author:
    def __init__(self, *args):
        args = list(args)
        if(len(args) == 0):
            self.name = "Default"
            self.list_books = []
        elif(len(args) == 1):
            self.name = args[0]
            self.list_books = []
        else:
            self.name = args.pop(0)
            self.list_books = args



    def changeName(self, name):
        self.name = name


    def addBooks(self, *args):
        self.list_books += list(args)


    def printDetails(self):
        print("Author Name:", self.name)
        print('-----------')
        print("List of Books:")
        for books in self.list_books:
            print(books)



auth1 = Author('Humayun Ahmed')
auth1.addBooks('Deyal', 'Megher Opor Bari')
auth1.printDetails()
print('===================')
auth2 = Author()
print(auth2.name)
auth2.changeName('Mario Puzo')
auth2.addBooks('The Godfather', 'Omerta', 'The Sicilian')
print('===================')
auth2.printDetails()
print('===================')
auth3 = Author('Paolo Coelho', 'The Alchemist', 'The Fifth Mountain')
auth3.printDetails()
```

```
Author Name: Humayun Ahmed
-----------
List of Books:
Deyal
Megher Opor Bari
===================
Default
===================
Author Name: Mario Puzo
-----------
```

```
List of Books:
The Godfather
Omerta
The Sicilian
===================
Author Name: Paolo Coelho
-----------
List of Books:
The Alchemist
The Fifth Mountain
```

```
[ ]:
```