



Inspiring Excellence

Course Title: Programming Language II

Course Code: CSE 111

Lab Assignment no: 4

Question 1

Suppose your little sibling wants your help to check his math homework. He is done with his homework but wants you to see if all his results are correct. Since the student with all correct results gets 3 stars. However, you want your brother to check this on his own. So, you design a calculator for him in python. You could have given your scientific calculator but you wanted to give him a basic calculator and also wanted to see if you can even design one.

Subtasks:

1. Create a class called Calculator.
2. Your class shall have 1 constructor and 4 methods, namely add, subtract, multiply and divide.
3. Now, create an object of your class. After creating an object, it should print "Let's Calculate!"
4. Then take 3 inputs from the user: first value, operator, second value
5. Now based on the given operator, call the required method and print the result.

Sample Input:

1
+
2

Sample Output:

Let's Calculate!
Value 1: 1
Operator: +
Value 2: 2
Result: 3

Question 2

Write a class called **Customer** with the required constructor and methods to get the following output.

Subtasks:

1. Create a class called Customer.
2. Create the required constructor.
3. Create a method called **greet** that works if no arguments are passed or if one argument is passed. (*Hint: You may need to use the keyword NONE*)
4. Create a method called **purchase** that can take as many arguments as the user wants to give.

[You are not allowed to change the code below]

<p># Write your codes for subtasks 1-4 here.</p> <pre>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")</pre>	<p>OUTPUT:</p> <pre>Hello! Sam, you purchased 3 item(s): chips chocolate orange juice ----- Hello David! David, you purchased 1 item(s): orange juice</pre>
--	--

Question 3

The Giant Panda Protection and Research Center in the Sichuan province of southwest China, actually employs a category of workers known as panda nannies. The primary responsibility is to play with adorable panda cubs and name them, determine gender, keep track of their age and hours they sleep. So being a programmer panda nanny, you will create a code that will do all these works for you.

1. Create a class named **Panda** and also write the constructor.
2. Access the instance attributes and print them in the given format.
3. Call instance methods to keep track of their daily hours of sleep.
4. Suppose consulting with other panda nannies you have set some criteria based on which you will make their diet plans. The criteria are:
 - ** Mixed Veggies for pandas having 3 to 5 hours (included) of sleep daily.
 - ** Eggplant & Tofu for pandas having 6 to 8 hours (included) of sleep daily.
 - ** Broccoli Chicken for pandas having 9 to 11 hours (included) of sleep daily.
 - ** Lastly if no arguments are passed then just give it bamboo leaves.

Now handle this problem modifying the method designed to keep track of their daily hours of sleep and determine diet plan using method overloading.

[You are not allowed to change the code below]

#Write your code here for subtasks 1-4.

```
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())
```

OUTPUT:

```
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
```

Question 4

Analyze the given code below to write **Cat** class to get the output as shown.

Hints:

- Remember, the constructor is a special method. Here, you have to deal with constructor overloading which is similar to method overloading.
- You may need to use the keyword *None*
- Your class should have 2 variables

[You are not allowed to change the code below]

#Write your code here	OUTPUT
<pre>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()</pre>	<pre>White cat is sitting Black cat is sitting Brown cat is jumping Red cat is purring Blue cat is sitting Purple cat is jumping</pre>

Question 5

Implement the design of the **Student** class so that the following output is produced:

Driver Code	Output
<pre># Write your code here 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()</pre>	<pre>----- 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</pre>

Question 6

Design a “**Vehicle**” class. A vehicle assumes that the whole world is a 2-dimensional graph paper. It maintains its x and y coordinates (both are integers). Any new object created of the Vehicle class will always start at the coordinates (0,0).

It must have methods to move up, down, left, right and a print_position() method for printing the current coordinate.

Note: All moves are 1 step. That means a single call to any move method changes the value of either x or y or both by 1.

[You are not allowed to change the code below]

<i># Write your class here</i>	<i>OUTPUT</i>
<pre>car = Vehicle() car.print_position() car.moveUp() car.print_position() car.moveLeft() car.print_position() car.moveDown() car.print_position() car.moveRight()</pre>	<pre>(0,0) (0,1) (-1,1) (-1,0)</pre>

Question 7

Design the **Programmer** class such a way so that the following code provides the expected output.

Hint:

- Write the constructor with appropriate printing and multiple arguments.
- Write the addExp() method with appropriate printing and argument.
- Write the prinDetails() method

[You are not allowed to change the code below]

Write your code here.

```
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()
```

OUTPUT:

```
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.
```

Question 8

Design the **Student** class such a way so that the following code provides the expected output.

Hint:

- Write the constructor with appropriate default value for arguments.
- Write the `dailyEffort()` method with appropriate argument.
- Write the `printDetails()` method. For printing suggestions check the following instructions.
 - If `hour <= 2` print 'Suggestion: Should give more effort!'
 - If `hour <= 4` print 'Suggestion: Keep up the good work!'
 - Else print 'Suggestion: Excellent! Now motivate others.'

[You are not allowed to change the code below]

Write your code here.

```
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()
```

OUTPUT:

```
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.
```


Question 9

Implement the design of the **Patient** class so that the following output is produced:

[You are not allowed to change the code below]

Write your code here.

```
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("=====")
```

OUTPUT:

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

Question 10

Implement the design of the **Avengers** class so that the following output is produced:

[You are not allowed to change the code below]

Write your code here.

```
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("=====")
```

OUTPUT:

```
=====
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
=====
```

Question 11

Design the **Shinobi** class such a way so that the following code provides the expected output.

Hint:

- Write the constructor with appropriate default value for arguments. Set the initial salary and mission to 0.
- Write the changeRank() method with appropriate argument.
- Write the calSalary() method with appropriate argument. Check the following suggestions
 - Update the number of mission from the given argument.
 - If rank == 'Genin' then salary = #mission * 50
 - If rank == 'Chunin' then salary = #mission * 100
 - else salary = #mission * 500
- Write the printInfo() method with appropriate printing.

[You are not allowed to change the code below]

Write your code here.

```
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()
```

OUTPUT:

```
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
```

Question 12

Design the **ParcelKoro** class such a way so that the following code provides the expected output.

Hint: `total_fee = (total_weight * 20) + location_charge`.

Note: For the method calculate fee: if the delivery location is not given, the `location_charge` will be 50 taka or else 100 taka. Also, while calculating total fee, if the product weight is 0 the `total_fee` should also be 0.

Assume only these 3 ways you can create an object of a class.

[You are not allowed to change the code below]

Write your code here.

```
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()
```

OUTPUT:

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

Question 13

Implement the design of the Batsman class so that the following output is produced:

Hint: Batting strike rate (s/r) = runsScored / ballsFaced x 100.

Driver Code	Output
<pre># Write your code here 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())</pre>	<pre>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</pre>

Question 14

Implement the design of the EPL_Team class so that the following output is produced:

Driver Code	Output
<pre># Write your code here 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())</pre>	<pre>===== 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</pre>

Question 15

Implement the design of the **Account** class so that the following output is produced:

Driver Code	Output
# Write your code here 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)	Default Account 0.0 ----- Oliver 10000.0 ----- Liam 0.0 ----- 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

Question 16

Implement the design of the **Author** class so that the following output is produced:

Driver Code	Output
<pre># Write your code here 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()</pre>	<pre>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</pre>