Inspiring Excellence

Course Code:	**CSE111**
Course Title:	**Programming Language II**
Homework No:	**03**
Topic:	**OOP(Classes and objects)**
Submission Type:	**Hard Copy (Only submit the part of the code that you have been instructed to write. DO NOT write any given code.)**
Resources:	**1. Class lectures** **2. BuX lectures** **a. English: https://shorturl.at/dhjAZ** **b. Supplementary: https://shorturl.at/wMPRU**

TASK 1

Design the **CellPackage** class and write suitable driver code to produce the given output:
Subtasks:

- (#1) ***Assign*** the arguments into appropriate attributes: **data**, **talk_time**, **messages**, **cashback**, **validity** and **price** via a parameterized constructor. All the attributes should be of `int` data type. Note that **data** is stored in *Megabytes* (*1 GB = 1024 MB*) and the **cashback** amount is calculated from a *percentage value*.
- (#2,3,4) ***Implement*** driver code to display all the information of a package. ***Check*** if any particular attribute does not exist (is equal to 0), do not print that attribute. Attributes **validity** and **price** are always printed.

```
# Subtask 1: Write the CellPackage Class

pkg = CellPackage(150, '6 GB', 99, 20, '7%', 7)
print('============= Package 1 =============')
# Subtask 2: Check each attribute and print


pkg2 = CellPackage(700, '35 GB', 700, 0, '10%', 30)
print('============= Package 2 =============')
# Subtask 3: Check each attribute and print


pkg4 = CellPackage(120, '0 GB', 190, 0, '0%', 10)
print('============= Package 3 =============')
# Subtask 4: Check each attribute and print
```

Expected Output:

```
============= Package 1 =============
Data = 6144 MB
Talktime = 99 Minutes
SMS/MMS = 20
Validity = 7 Days
--> Price = 150 tk
Buy now to get 10 tk cashback.
============= Package 2 =============
Data = 35840 MB
Talktime = 700 Minutes
Validity = 30 Days
```

```
--> Price = 700 tk
Buy now to get 70 tk cashback.
============= Package 3 =============
Talktime = 190 Minutes
Validity = 10 Days
--> Price = 120 tk
```

TASK 2

Design a class called **Pokemon** using a parameterized constructor so that after executing the following line of code the desired result shown in the output box will be printed.

The first object along with print has been done for you, you also need to create other objects and print accordingly to get the output correctly.

Subtasks:

1. Design the **Pokemon** class using a parameterized constructor.
   The 5 values that are being passed through the constructor are respectively:
   1. pokemon 1 name,
   2. pokemon 2 name,
   3. pokemon 1 power,
   4. pokemon 2 power and
   5. damage rate
2. Create an object named **team_bulb** and pass the values `'bulbasaur', 'squirtle', 80, 70, 9` respectively.
3. Use print statements accordingly to print the desired result of **team_bulb**.

[You are not allowed to change the code below]

```
# Write your code for class here

team_pika = Pokemon('pikachu', 'charmander', 90, 60, 10)
print('=======Team 1=======')
print('Pokemon 1:',team_pika.pokemon1_name,
team_pika.pokemon1_power)
print('Pokemon 2:',team_pika.pokemon2_name,
team_pika.pokemon2_power)
pika_combined_power = (team_pika.pokemon1_power +
team_pika.pokemon2_power) * team_pika.damage_rate
```

```
print('Combined Power:', pika_combined_power)

# Write your code for subtask 2 and 3 here
```

Expected Output:

```
=======Team 1=======
Pokemon 1: pikachu 90
Pokemon 2: charmander 60
Combined Power: 1500
=======Team 2=======
Pokemon 1: bulbasaur 80
Pokemon 2: squirtle 70
Combined Power: 1350
```

TASK 3

Part A

Write the **box** class so that the given drive code gives the expected output.

[You are not allowed to change the code below]

```
# Write your class code here

print("Box 1")
b1 = box([10,10,10])
print("=========================")
print("Height:", b1.height)
print("Width:", b1.width)
print("Breadth:", b1.breadth)
volume = b1.height * b1.width *
b1.breadth
print(f"Volume of the box is {volume}
cubic units.")
print("-------------------------")
print("Box 2")
b2 = box((30,10,10))
print("=========================")
print("Height:", b2.height)
print("Width:", b2.width)
print("Breadth:", b2.breadth)
```

Expected Output:

```
Box 1
Creating a Box!
=========================
Height: 10
Width: 10
Breadth: 10
Volume of the box is 1000
cubic units.
-------------------------
Box 2
Creating a Box!
=========================
Height: 30
Width: 10
Breadth: 10
Volume of the box is 3000
cubic units.
Updating Box 2!
```

```
volume = b2.height * b2.width *
b2.breadth
print(f"Volume of the box is {volume}
cubic units.")
b2.height = 300
print("Updating Box 2!")
print("Height:", b2.height)
print("Width:", b2.width)
print("Breadth:", b2.breadth)
volume = b2.height * b2.width *
b2.breadth
print(f"Volume of the box is {volume}
cubic units.")
print("-------------------------")
print("Box 3")
b3 = b2
print("Height:", b3.height)
print("Width:", b3.width)
print("Breadth:", b3.breadth)
volume = b3.height * b3.width *
b3.breadth
print(f"Volume of the box is {volume}
cubic units.")
```

```
Height: 300
Width: 10
Breadth: 10
Volume of the box is 30000
cubic units.
-------------------------
Box 3
Height: 300
Width: 10
Breadth: 10
Volume of the box is 30000
cubic units.
```

Part B

After the given driver code, if we run the following lines of code:

```
one = (b3 == b2)

b3.width = 100
two = (b3 == b2)
```

1. What will be the values of the variables `one` and `two`? Explain your answer briefly in text.
2. What will be the value of `b2.width`? Has that value changed since the driver code ran? If yes, explain why in brief text.

TASK 4

Read the following **Vector3D** class that represents a vector in 3D space. The x-axis, y-axis, and z-axis components of the vector are represented by the attributes *x, y,* and *z* respectively.

[You are not allowed to change the code below]

```
class Vector3D:
    def __init__(self, x, y, z):
        self.x = x
        self.y = y
        self.z = z
        print(f'Vector <{self.x}, {self.y},
{self.z}> has been created.')

# Write your driver code here
```

Your task is to write the driver code that:

- Creates 2 **Vector3D** objects. The first one is given as $V_1 =< 2, \ -3, \ 1 >$ and the second one is given as $V_2 =< -1, \ 4, \ 0 >$.
- Prints their components and magnitude as shown in the output. The magnitude of a vector $< a, \ b, \ c >$ is calculated as $| < a, \ b, \ c > | = \sqrt{a^2 + b^2 + c^2}$.
- Finds the **dot product** of the 2 Vectors. Dot product of 2 vectors $< a_1, \ a_2, \ a_3 >$ and $< b_1, \ b_2, \ b_3 >$ is calculated as
  $< a_1, \ a_2, \ a_3 > \ \cdot \ < b_1, \ b_2, \ b_3 > \ = a_1 b_1 + a_2 b_2 + a_3 b_3$.
- Finds the **Cross product** of the 2 Vectors. The cross product of 2 vectors $< a_1, \ a_2, \ a_3 >$ and $< b_1, \ b_2, \ b_3 >$ creates ***a new Vector3D Object*** which is calculated as
  $< a_1, \ a_2, \ a_3 > \ \times \ < b_1, \ b_2, \ b_3 > \ = \ < a_2 b_3 - a_3 b_2, \ a_3 b_1 - a_1 b_3, \ a_1 b_2 - a_2 b_1 >$
  .
- Generates the output as given below.
- Your program should run for any two 3D vectors.

Expected Output:

```
Vector <2, -3, 1> has been created.
Vector <-1, 4, 0> has been created.
Magnitude of the first vector = 3.7416573867739413
Magnitude of the second vector = 4.123105625617661
Dot product of the two vectors = -14
Vector <-4, -1, 5> has been created.
Cross product of the two vectors = <-4, -1, 5>
```

TASK 5

Design the **Order** class so that it generates the expected output for the given Driver code. The **Order** class has an attribute named ***items***, which is a list that is created in the following pattern:

[i_1, q_1, p_1, i_2, q_2, p_2, i_3, q_3, p_3, . . .].

Here, ***i_x***, ***q_x,*** and ***p_x*** refer to the **item name**, **quantity ordered,** and **subtotal price** of the **x-th ordered item** respectively.

[You are not allowed to change the code below]

```
# Write your class code here

menu = {
    'Chicken_Cheeseburger' : 249,
    'Mega_Cheeseburger' : 289,
    'Fries' : 139,
    'Hot_Wings' : 99,
    'Rice_Bowl' : 299,
    'Soft_Drinks' : 50
}

order1 = Order(menu, "Chicken_Cheeseburger-2, Fries-3,
Soft_Drinks-3")
print(order1.items)
print()

print('-'*35)
print('Item           x Quantity :   Price')
print('--------------   --------   -------')
```

```
index = 0
total = 0
while index < len(order1.items):
  item = order1.items[index]
  quantity = order1.items[index+1]
  price = order1.items[index+2]

  print(f'{item:20} x {quantity:2} : {price:7.2f}')
  total += price
  index += 3 # Going to next item

print('-'*35)
print(f'Total:                        {total:7.2f}')
print('-'*35)
```

Expected Output:

```
['Chicken_Cheeseburger', 2, 498, 'Fries', 3, 417, 'Soft_Drinks', 3,
150]

-----------------------------------
Item           x Quantity :   Price
--------------   --------   -------
Chicken_Cheeseburger x  2 :  498.00
Fries                x  3 :  417.00
Soft_Drinks          x  3 :  150.00
-----------------------------------
Total:                       1065.00
-----------------------------------
```