

| **Course Code:** | **CSE111** |
| --- | --- |
| **Course Title:** | **Programming Language II** |
| **Classwork No:** | **04** |
| **Topic:** | **OOP (Instance method and overloading)** |
| **Number of Tasks:** | **7** |

**Task 1**

Design the **Customer** class with the necessary properties so that the following output is produced.

[**Hint:**

* If the visitor’s age is greater than 10, then the ticket price is 100 taka. Otherwise, 50 taka.
* A customer can’t buy more than 3 tickets.]

| **Driver Code** | **Given Output** |
| --- | --- |
| **print('1-------------------------')**  **customer1 = Customer()**  **print('2-------------------------')**  **customer1.buyTicket('Bob', 23)**  **customer1.buyTicket('Henry', 7)**  **customer1.buyTicket('Alexa', 30)**  **customer1.buyTicket('Jonas', 43)**  **print('3-------------------------')**  **customer1.showDetails()**  **print('4-------------------------')**  **customer2 = Customer()**  **print('5-------------------------')**  **customer2.buyTicket('Harry', 60)**  **customer2.buyTicket('Tomas', 28)**  **print('6-------------------------')**  **customer2.showDetails()** | **1--------------------------**  **Welcome to ABC Memorial Park**  **2--------------------------**  **Successfully purchased a ticket for Bob!**  **Successfully purchased a ticket for Henry!**  **Successfully purchased a ticket for Alexa!**  **You can't buy more than 3 tickets**  **3--------------------------**  **Amount of tickets: 3**  **Total price: 250 Taka**  **4--------------------------**  **Welcome to ABC Memorial Park**  **5--------------------------**  **Successfully purchased a ticket for Harry!**  **Successfully purchased a ticket for Tomas!**  **6--------------------------**  **Amount of tickets: 2**  **Total price: 200 Taka** |

**Task 2**

You are given a string containing a sentence. Your task is to implement the **SentenceAnalyzer** class with constructor overloading and method overloading to analyze the given sentence.

| **Driver Code** | **Given Output** |
| --- | --- |
| **analyzer1 = SentenceAnalyzer()**  **analyzer1.set\_sentence("That's an easy one")**  **print("1--------------------------")**  **analyzer1.get\_word\_count()**  **print("2--------------------------")**  **analyzer2 = SentenceAnalyzer("Like I said it's easy")**  **print("3--------------------------")**  **analyzer2.get\_word\_count()**  **print("4--------------------------")**  **analyzer2.get\_word\_count(4)**  **print("5--------------------------")**  **analyzer1.get\_word\_count(5)** | **1--------------------------**  **Number of words in the sentence: 4 2--------------------------**  **3--------------------------**  **Number of words in the sentence: 5**  **4--------------------------**  **Number of words with 4 characters in the sentence: 4**  **5--------------------------**  **Number of words with 5 characters in the sentence: 0** |

**Task 3**

Design the **Student** class with the necessary properties so that the given output is produced for the provided driver code. Use constructor overloading and method overloading where necessary.

***Hint:***

* ***A student having cgpa>=3.5 and credit>10 is eligible for scholarship.*** 
  + ***A student having cgpa >=3.7 is eligible for Merit based scholarship***
  + ***A student with cgpa>=3.5 but <3.7 is eligible for Need-based scholarship.***

| **Driver Code** | **Given Output** |
| --- | --- |
| **print('--------------------------')**  **std1 = Student("Alif", 3.99, 12)**  **print('--------------------------')**  **std1.checkScholarshipEligibility()**  **print('--------------------------')**  **std1.showDetails()**  **print('--------------------------')**  **std2 = Student("Mim", 3.4)**  **std3 = Student("Henry", 3.5, 15,"BBA")**  **print('--------------------------')**  **std2.checkScholarshipEligibility()**  **print('--------------------------')**  **std3.checkScholarshipEligibility()**  **print('--------------------------')**  **std2.showDetails()**  **print('--------------------------')**  **std3.showDetails()**  **print('--------------------------')**  **std4 = Student("Bob", 4.0, 6, "CSE")**  **print('--------------------------')**  **std4.checkScholarshipEligibility()**  **print('--------------------------')**  **std4.showDetails()** | **--------------------------**  **--------------------------**  **Alif is eligible for Merit-based scholarship.**  **--------------------------**  **Name: Alif**  **Department: CSE**  **CGPA: 3.99**  **Number of Credits: 12**  **Scholarship Status: Merit-based scholarship**  **--------------------------**  **--------------------------**  **Mim is not eligible for scholarship.**  **--------------------------**  **Henry is eligible for Need-based scholarship.**  **--------------------------**  **Name: Mim**  **Department: CSE**  **CGPA: 3.4**  **Number of Credits: 9**  **Scholarship Status: No scholarship**  **--------------------------**  **Name: Henry**  **Department: BBA**  **CGPA: 3.5**  **Number of Credits: 15**  **Scholarship Status: Need-based scholarship**  **--------------------------**  **--------------------------**  **Bob is not eligible for scholarship.**  **--------------------------**  **Name: Bob**  **Department: CSE**  **CGPA: 4.0**  **Number of Credits: 6**  **Scholarship Status: No scholarship** |

**Task 4**

Write the **Author** class with the required properties so that the given output is produced for the provided driver code:

| **Driver Code** | **Output** |
| --- | --- |
| ***# Write your code here***  **# Do not change the following lines of code.**  **a1 = Author()**  **print("===============================")**  **a1.addBook("Ice", "Science Fiction")**  **print("===============================")**  **a1.setName("Anna Kavan")**  **a1.addBook("Ice", "Science Fiction")**  **a1.printDetail()**  **print("===============================")**  **a2 = Author("Humayun Ahmed")**  **a2.addBook("Onnobhubon","Science Fiction")**  **a2.addBook("Megher Upor Bari", "Horror")**  **print("===============================")**  **a2.printDetail()**  **a2.addBook("Ireena", "Science Fiction")**  **print("===============================")**  **a2.printDetail()**  **print("===============================")** | **=================================**  **A book can not be added without author name**  **=================================**  **Number of Book(s): 1**  **Author Name: Anna Kavan**  **Science Fiction: Ice**  **=================================**  **=================================**  **Number of Book(s): 2**  **Author Name: Humayun Ahmed**  **Science Fiction: Onnobhubon**  **Horror: Megher Upor Bari**  **=================================**  **Number of Book(s): 3**  **Author Name: Humayun Ahmed**  **Science Fiction: Onnobhubon, Ireena**  **Horror: Megher Upor Bari**  **=================================** |

**Task 5**

Using the **TaxiLagbe** app, users can share a single taxi with multiple people.

**Implement** the design of the **TaxiLagbe** class with the necessary properties so that the given output is produced for the provided driver code:

**[Hint:**

1. Each taxi can carry a maximum of 4 passengers

2. The addPassenger() method takes the last name of the passenger and ticket fare for that person in an underscore (\_)-separated string.**]**

| **Driver Code** | **Output** |
| --- | --- |
| ***# Write your code here***  **# Do not change the following lines of code.**  **taxi1 = TaxiLagbe('1010-01', 'Dhaka')**  **print('-------------------------------')**  **taxi1.addPassenger('Walker\_100', 'Wood\_200','Matt\_100')**  **taxi1.addPassenger('Wilson\_105')**  **print('-------------------------------')**  **taxi1.printDetails()**  **print('-------------------------------')**  **taxi1.addPassenger('Karen\_200')**  **print('-------------------------------')**  **taxi1.printDetails()**  **print('-------------------------------')**  **taxi2 = TaxiLagbe('1010-02', 'Khulna')**  **taxi2.addPassenger('Ronald\_115', 'Parker\_215')**  **print('-------------------------------')**  **taxi2.printDetails()** | **--------------------------------------**  **Dear Walker! Welcome to TaxiLagbe.**  **Dear Wood! Welcome to TaxiLagbe.**  **Dear Matt! Welcome to TaxiLagbe.**  **Dear Wilson! Welcome to TaxiLagbe.**  **--------------------------------------**  **Trip info for Taxi number: 1010-01**  **This taxi can only cover the Dhaka area.**  **Total passengers: 4**  **Passenger lists:**  **Walker, Wood, Matt, Wilson**  **Total collected fare: 505 Taka**  **--------------------------------------**  **Taxi Full! No more passengers can be added.**  **--------------------------------------**  **Trip info for Taxi number: 1010-01**  **This taxi can only cover the Dhaka area.**  **Total passengers: 4**  **Passenger lists:**  **Walker, Wood, Matt, Wilson**  **Total collected fare: 505 Taka**  **--------------------------------------**  **Dear Ronald! Welcome to TaxiLagbe.**  **Dear Parker! Welcome to TaxiLagbe.**  **--------------------------------------**  **Trip info for Taxi number: 1010-02**  **This taxi can only cover the Khulna area.**  **Total passengers: 2**  **Passenger lists:**  **Ronald, Parker**  **Total collected fare: 330 Taka** |

**Task 6**

| **1** | **class Scope:** |
| --- | --- |
| **2** | **def \_\_init\_\_(self):** |
| **3** | **self.x, self.y = 1, 100** |
| **4** | **def met1(self):** |
| **5** | **x = 3** |
| **6** | **x = self.x + 1** |
| **7** | **self.y = self.y + self.x + 1** |
| **8** | **x = self.y + self.met2() + self.y** |
| **9** | **print(x, self.y)** |
| **10** | **def met2(self):** |
| **11** | **y = 0** |
| **12** | **print(self.x, y)** |
| **13** | **self.x = self.x + y** |
| **14** | **self.y = self.y + 200** |
| **15** | **return self.x + y** |

| **Write the output of the following code:**  **q2 = Scope()**  **q2.met1()**  **q2.met2()**  **q2.met1()**  **q2.met2()** | **x** | **y** |
| --- | --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Task 7**

| **1** | **class Test3:** |
| --- | --- |
| **2** | **def \_\_init\_\_(self):** |
| **3** | **self.sum, self.y = 0, 0** |
| **4** | **def methodA(self):** |
| **5** | **x, y = 2, 3** |
| **6** | **msg = [0]** |
| **7** | **msg[0] = 3** |
| **8** | **y = self.y + msg[0]** |
| **9** | **self.methodB(msg)** |
| **10** | **x = self.y + msg[0]** |
| **11** | **self.sum = x + y + msg[0]** |
| **12** | **print(x, y, self.sum)** |
| **13** | **def methodB(self, mg2, mg1=6):** |
| **14** | **x = 0** |
| **15** | **self.y = self.y + mg2[0]** |
| **16** | **x = x + 33 + mg1** |
| **17** | **self.sum = self.sum + x + self.y** |
| **18** | **mg2[0] = self.y + mg1** |
| **19** | **mg1 = mg1 + x + 2** |
| **20** | **print(x, self.y, self.sum)** |

| **Write the output of the following code:**  **t3 = Test3()**  **t3.methodA()**  **t3.methodA()**  **t3.methodA()**  **t3.methodA()** | **x** | **y** | **sum** |
| --- | --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |