BRAC UNIVERSITY
Department of Computer Science and Engineering

Examination: Final | **Semester**: Spring 2023
Duration: 90 Minutes | CSE 111: Programming Language II | **Full Marks**: 30
No. of Questions: 3 | **No. of Pages**: 3

Name: (Please write in CAPITAL LETTERS)	ID:	Section:

B

✓ Use the back **part** of the answer script for rough work. **No washroom breaks.**
✓ At the end of the exam, put the question **paper** inside the answer script and **return both**.

Question – 1: CO4 [10 Points]

```
1  class A:
2      temp = 3
3      def __init__(self):
4          self.y = A.temp - 2
5          self.sum = self.temp + 1
6          A.temp += 3
7          self.methodA(6, 3)
8      def methodA(self, m,  n):
9          x = self.y + 4 + n
10         self.sum = x + self.temp
11         self.y = self.y + m + (A.temp)
12         print(x, self.y, self.sum)
13 class B(A):
14     def __init__(self, obj=None):
15         super().__init__()
16         self.temp = self.temp + A.temp
17         self.sum = 5 + self.temp + A.temp
18     def methodA(self, m,  n, x=4):
19         self.y = self.y + n + (A.temp)
20         x = x + 3 + m
21         self.sum = self.sum + x + self.temp
22         print(x, self.y, self.sum)
23     def methodB(self, m, n):
24         y = self.temp + self.y + n
25         A.temp = m + self.y + n
26         super().methodA(n, m)
27         self.methodA(n, m)
28         self.sum = self.y + y + A.temp
29         print(self.temp , y, self.sum)
```

Illustrate the output of the following statements:

```
b1 = B()
b1.methodB(4, 5)
```

Output:

Out1	Out2	Out3
		54
	27	103

Question 2: CO5 [10 Points]

Design the **ChefsCounter** class with necessary properties such that the following output is produced for the given driver code.
[**Hint:** Unless mentioned explicitly, a branch can take 5 reservations by default.]

```
print("==============1============")
branch1 = ChefsCounter("Gulshan")
print("==============2============")
branch1.reserve("Sam", "Paul")
print("==============3============")
branch1.reservation_info()
print("==============4============")
branch1.reserve("John", "Robin", "Billy",
"Robert")
print("==============5============")
branch1.reservation_info()
print("==============6============")
branch2 = ChefsCounter("Dhanmondi",7)
print("==============7============")
branch2.reserve("Ben", "Alice", "Fred")
print("==============8============")
branch2.reservation_info()
print("==============9============")
branch2.reserve("Tom", "Ken", "Garet",
"Miles", "Taylor")
print("==============10============")
branch2.reservation_info()
print("==============11============")
branch3 = ChefsCounter.createNewBranch("100
feet")
print("==============12============")
branch3.reserve("Harry", "Bob", "Jenny")
print("==============13============")
branch3.reservation_info()
print("==============14============")
print("Reservation Information of All
Branches:",ChefsCounter.reservation)
```

Output:

```
==============1============
The Gulshan branch of Chef's Counter is
open for reservation!
==============2============
==============3============
Customers who reserved in Gulshan branch:
Sam, Paul
==============4============
Sorry Robert, 5 people already made a
reservation in this branch.
==============5============
Customers who reserved in Gulshan branch:
Sam, Paul, John, Robin, Billy
==============6============
The Dhanmondi branch of Chef's Counter is
open for reservation!
==============7============
==============8============
Customers who reserved in Dhanmondi
branch:
Ben, Alice, Fred
==============9============
Sorry Taylor, 7 people already made a
reservation in this branch.
==============10============
Customers who reserved in Dhanmondi
branch:
Ben, Alice, Fred, Tom, Ken, Garet, Miles
==============11============
The 100 feet branch of Chef's Counter is
open for reservation!
==============12============
==============13============
Customers who reserved in 100 feet branch:
Harry, Bob, Jenny
==============14============
Reservation Information of All Branches:
{'Gulshan': ['Sam', 'Paul', 'John',
'Robin', 'Billy'], 'Dhanmondi': ['Ben',
'Alice', 'Fred', 'Tom', 'Ken', 'Garet',
'Miles'], '100 feet': ['Harry', 'Bob',
'Jenny']}
```

Question 3: CO5 [10 Points]

Implement the "**BusTicket**" class derived from the "**Ticket**" class with the necessary properties so that the following output is produced for the given code.
[**Hint:**

1. Ticket ID is calculated by concatenating the bus name, a "-", and number of tickets being created.

2. Fare calculation needs to be done inside the "**`calculate_fare()`**" method and the calculated fare must be assigned to the "**`price`**" variable inherited from the Ticket class.

3. Make sure you reuse the parent class's code wherever possible.]

```
class Ticket:
  route_distance = {"Route A":400, "Route B":425,
"Route C":350}
  fare_per_km = 20

  def __init__(self, route, journeyDate, price = 0):
    self.route = route
    self.journeyDate = journeyDate
    self.__price = price
  def setPrice(self, price):
    self.__price = price
  def getPrice(self):
    return self.__price
  def ticket_details(self):
    return f"Route: {self.route}\nJourney Date:
{self.journeyDate}"

#Driver Code

ticket1 = BusTicket("Route A", "30 April, 2023",
"Nabil Enterprise", "F2")
print("Total ticket(s):", BusTicket.total_tickets)
print("1==========================")
ticket1.calculate_fare()
print("2==========================")
ticket1.ticket_details()
print("3==========================")
ticket1.make_payment()
print("4==========================")
ticket1.ticket_details()
print("5==========================")
ticket2 = BusTicket("Route C", "26 April, 2023",
"Hanif Enterprise", "A2")
print("Total ticket(s):", BusTicket.total_tickets)
print("6==========================")
ticket2.calculate_fare()
print("7==========================")
ticket2.make_payment()
print("8==========================")
ticket2.ticket_details()
```

Output:

```
Total ticket(s): 1
1==========================
Ticket fare is calculated
successfully.
2==========================
Ticket ID: Nabil Enterprise-1
Route: Route A
Journey Date: 30 April, 2023
Bus Name: Nabil Enterprise
Seat No: F2
Price(tk): 8000
Status: Not Paid
3==========================
Payment successful.
4==========================
Ticket ID: Nabil Enterprise-1
Route: Route A
Journey Date: 30 April, 2023
Bus Name: Nabil Enterprise
Seat No: F2
Price(tk): 8000
Status: Paid
5==========================
Total ticket(s): 2
6==========================
Ticket fare is calculated
successfully.
7==========================
Payment successful.
8==========================
Ticket ID: Hanif Enterprise-2
Route: Route C
Journey Date: 26 April, 2023
Bus Name: Hanif Enterprise
Seat No: A2
Price(tk): 7000
Status: Paid
```