Lab Assignment 04

|  |  |
| --- | --- |
| **Course Code:** | **CSE111** |
| **Course Title:** | **Programming Language II** |
| **Topic:** | **Constructor, Constructor Overloading and Multiclass Problem** |
| **Number of Tasks:** | **11 (Coding: 08, Tracing: 03)** |

# Task 1

Design the **Student** class in such a way that it produces the following output.

|  |  |
| --- | --- |
| **Driver Code** | **Expected Output** |
| public class StudentTester{  public static void main(String[] args){  Student s1 = new Student("Harry", "CSE");  System.*out*.println(s1.name);  s1.updateName("Harry Potter");  System.*out*.println(s1.name);  System.*out*.println(s1.prog);  s1.updateProgram("CS");  String prog = s1.accessProgram();  System.*out*.println(prog);  }  } | Harry  Harry Potter  CSE  CS |

# 

# Task 2

Design the **Toy** class in such a way that it produces the following output

|  |  |
| --- | --- |
| **Driver Code** | **Expected Output** |
| public class ToyTester{  public static void main(String[] args){  Toy t1 = new Toy("Car", 230);  System.out.println("1=================");  t1.updatePrice(250);  System.out.println("2=================");  System.out.println(t1.name);  t1.showPrice();  System.out.println("3=================");  Toy t2 = new Toy("Robot", 450);  System.out.println("4=================");  t2.updateName("Autobot");  System.out.println("5=================");  System.out.println(t2.name);  t2.showPrice();  }  } | A new toy has been made!  1=================  2=================  Car  price: 250 Taka  3=================  A new toy has been made!  4=================  Changing old name: Robot  new name: Autobot  5=================  Autobot  price: 450 Taka |

# Task 3

Design the **Shape2D** class in such a way that it produces the following output.

|  |  |
| --- | --- |
| **Driver Code** | **Expected Output** |
| **public** **class** Shape2DTester {  **public** **static** **void** main(String[] args) {  Shape2D sq = **new** Shape2D();  System.***out***.println("---------1----------");  sq.area();  System.***out***.println("---------2----------");  Shape2D rectangle = **new** Shape2D(5,6);  System.***out***.println("---------3----------");  rectangle.area();  System.***out***.println("---------4----------");  Shape2D tri1 = **new** Shape2D(5,6,"Triangle");  System.***out***.println("---------5----------");  tri1.area();  System.***out***.println("---------6----------");  Shape2D tri2 = **new** Shape2D(5,6,7);  System.***out***.println("---------7----------");  tri2.area();  System.***out***.println("---------8----------");  }  } | A Square has been created with length: 5  ---------1----------  The area of the Square is: 25.0  ---------2----------  A Rectangle has been created with length: 5 and breadth: 6  ---------3----------  The area of the Rectangle is: 30.0  ---------4----------  A Triangle has been created with height: 5 and base: 6  ---------5----------  The area of the Triangle is: 15.0  ---------6----------  A Triangle has been created with the following sides: 5, 6, 7  ---------7----------  The area of the Triangle is: 14.69  ---------8---------- |

# 

# 

# 

# 

# Task 4

Write “**Student**“ class to show the following expected outputs

**Note:**

* A student can’t take any course until the CGPA is set.
* A student cannot take more than 4 courses.
* A student with CGPA below 3 cannot take more than 3 courses.

|  |  |
| --- | --- |
| **Driver Code** | **Expected Output** |
| public class StudentDriver {  public static void main(String[] args){  Student student1 = new Student(12345678);  System.out.println("1-----------------");  student1.addCourse("CSE110");  System.out.println("2-----------------");  student1.storeCG(2.5);  student1.addCourse(" ");  student1.addCourse("ENG101");  student1.showAdvisee();  System.out.println("3-----------------");  student1.removeAllCourse();  student1.showAdvisee();  System.out.println("4-----------------");  student1.storeID(54652365);  String[] courses = {"SOC101","CSE111","ENG102"};  student1.addCourse(courses);  student1.showAdvisee();  System.out.println("5-----------------");  student1.addCourse("CSE230");  student1.showAdvisee();  System.out.println("6-----------------");  Student student2 = new Student(975738383,3.7);  System.out.println("7-----------------");  String[] courses2 = {"CSE220","PHY112","MAT120","BUS101","CHN101"};  student2.addCourse(courses2);  student2.showAdvisee();  }  } | A student with ID 12345678 has been created.  1-----------------  Failed to add CSE110  Set CG first  2-----------------  Student ID: 12345678, CGPA: 2.5  Added courses are:  CSE110 ENG101  3-----------------  Student ID: 12345678, CGPA: 2.5  No courses added.  4-----------------  Student ID: 54652365, CGPA: 2.5  Added courses are:  SOC101 CSE111 ENG102  5-----------------  Failed to add CSE230  CG is low. Can't add more than 3 courses.  Student ID: 54652365, CGPA: 2.5  Added courses are:  SOC101 CSE111 ENG102  6-----------------  A student with ID 975738383 and cgpa 3.7 has been created.  7-----------------  Failed to add CHN101  Maximum 4 courses allowed.  Student ID: 975738383, CGPA: 3.7  Added courses are:  CSE220 PHY112 MAT120 BUS101 |

# Task 5

Design the **Triangle** Class that will produce the following output. We will consider both triangles to have the same sides if all sides are equal in the same orientation/sequence only.

Types of Triangle:

* Equilateral: When all sides in the same orientation are equal.
* Isosceles: When any two sides of a triangle in the same orientation are equal.
* Scalene: When all sides are of different lengths.

|  |  |
| --- | --- |
| **Driver Code** | **Output** |
| public class TriangleTester{  public static void main(String args[]){  Triangle t1 = new Triangle();  Triangle t2 = new Triangle();  Triangle t3 = new Triangle();  Triangle t4 = new Triangle();  t1.updateSides(4, 4, 4);  t2.updateSides(4, 5, 6);  t3.updateSides(4, 5, 6);  t4.updateSides(5, 4, 6);  t1.triangleDetails();  System.*out*.println("--------1--------");  System.*out*.println(t1.printTriangleType());  System.*out*.println("--------2--------");  t3.triangleDetails();  System.*out*.println(t3.printTriangleType());  System.*out*.println("--------3--------");  t4.triangleDetails();  System.*out*.println(t4.printTriangleType());  System.*out*.println("--------4--------");  t2.compareTrinagles(t3);  System.*out*.println("--------5--------");  t1.compareTrinagles(t2);  System.*out*.println("--------6--------");  t1 = t2;  t1.compareTrinagles(t2);  System.*out*.println("--------7--------");  t3.compareTrinagles(t4);  }  } | Three sides of the triangle are: 4, 4, 4  Perimeter: 12  --------1--------  This is an Equilateral Triangle.  --------2--------  Three sides of the triangle are: 4, 5, 6  Perimeter: 15  This is a Scalene Triangle.  --------3--------  Three sides of the triangle are: 5, 4, 6  Perimeter: 15  This is a Scalene Triangle.  --------4--------  Addresses are different but the sides of the triangles are equal.  --------5--------  Addresses, length of the sides and perimeter all are different.  --------6--------  These two triangle objects have the same address.  --------7--------  Only the perimeter of both triangles is equal. |

# 

# 

# Task 6

Write the **Teacher** and **Course** classes so that the TestTeacher class produces the outputs given. Hint: A teacher can add a maximum of 3 courses.

|  |  |
| --- | --- |
| **Driver Code** | **Output** |
| public class TestTeacher{  public static void main(String [] args){  Teacher t1 = new Teacher("Matin Saad Abdullah","MSA");  Teacher t2 = new Teacher("Mumit Khan","MMK");  Teacher t3 = new Teacher("Sadia Hamid Kazi","SKZ");  Course c1 = new Course("CSE 110");  Course c2 = new Course("CSE 111");  Course c3 = new Course("CSE 220");  Course c4 = new Course("CSE 221");  Course c5 = new Course("CSE 230");  Course c6 = new Course("CSE 310");  Course c7 = new Course("CSE 320");  Course c8 = new Course("CSE 340");  t1.addCourse(c1);  t1.addCourse(c2);  t2.addCourse(c3);  t2.addCourse(c4);  t2.addCourse(c5);  t3.addCourse(c6);  t3.addCourse(c7);  t3.addCourse(c8);  System.out.println("1========================");  t1.printDetail();  System.out.println("2========================");  t2.printDetail();  System.out.println("3========================");  t3.printDetail();  }  } | A new teacher has been created  A new teacher has been created  A new teacher has been created  1========================  Name: Matin Saad Abdullah  Initial: MSA  List of courses:  CSE 110  CSE 111  2========================  Name: Mumit Khan  Initial: MMK  List of courses:  CSE 220  CSE 221  CSE 230  3========================  Name: Sadia Hamid Kazi  Initial: SKZ  List of courses:  CSE 310  CSE 320  CSE 340 |

# 

# Task 7

Design the required class/es so that the following output is generated. Read the following description:

1. You may assume that to board a bus, a student must have the bus pass, and his/her destination must match the route of the bus.
2. Additionally, the default maximum capacity of the bus is 2.

|  |  |
| --- | --- |
| **Driver Code** | **Output** |
| public class BracuStudentTester {  public static void main(String[] args) {  BracuStudent st1 = new BracuStudent("Afif", "Mirpur");  System.out.println("1===============");  BracuStudent st2 = new BracuStudent("Shanto", "Motijheel");  BracuStudent st3 = new BracuStudent("Taskin", "Mirpur");  st1.showDetails();  st2.showDetails();  System.out.println("2===============");  st3.showDetails();  System.out.println("3===============");  BracuBus bus1 = new BracuBus("Mirpur");  BracuBus bus2 = new BracuBus("Azimpur", 5);  bus1.showDetails();  bus2.showDetails();  System.out.println("4===============");  st2.getPass();  st3.getPass();  System.out.println("5===============");  st2.showDetails();  st3.showDetails();  System.out.println("6===============");  bus1.board();  System.out.println("7===============");  bus1.board(st1, st2);  System.out.println("8===============");  st1.getPass();  st2.updateHome("Mirpur");  st1.showDetails();  st2.showDetails();  System.out.println("9===============");  bus1.board(st1);  bus1.board(st2, st3);  System.out.println("10===============");  bus1.showDetails();  }  } | 1===============  Student Name: Afif  Lives in Mirpur  Have Bus Pass? false  Student Name: Shanto  Lives in Motijheel  Have Bus Pass? false  2===============  Student Name: Taskin  Lives in Mirpur  Have Bus Pass? false  3===============  Bus Route: Mirpur  Passenger Count: 0 (Max: 2)  Passengers on Board:  Bus Route: Azimpur  Passenger Count: 0 (Max: 5)  Passengers on Board:  4===============  5===============  Student Name: Shanto  Lives in Motijheel  Have Bus Pass? true  Student Name: Taskin  Lives in Mirpur  Have Bus Pass? true  6===============  No passengers  7===============  You don't have a bus pass!  You got on the wrong bus!  8===============  Student Name: Afif  Lives in Mirpur  Have Bus Pass? true  Student Name: Shanto  Lives in Mirpur  Have Bus Pass? true  9===============  Afif boarded the bus.  Shanto boarded the bus.  Bus is full!  10===============  Bus Route: Mirpur  Passenger Count: 2 (Max: 2)  Passengers on Board:  Afif Shanto |

# Task 8

Design the **Student** and the **Usis** class so that the following output is produced.

Note:

* A student's email, password, and login status are null by default while creating an object of the Student class.
* Your code should satisfy the conditions mentioned in the output only.
* Usis class will have two instance variables: totalAdvisee and an array of Student type to store the student object. The array will be updated inside the advising() method only when the advising is successful.
* Usis can take at most 5 advisees.

|  |  |
| --- | --- |
| **Driver Code** | **Expected Output** |
| public class UsisTester {  public static void main(String[] args) {  Student rakib = new Student("Rakib", 12301455, "CSE");  Student roy = new Student("Roy", 12501345, "CS");  System.out.println("1\*\*\*\*\*\*\*\*\*\*\*\*\*");  Usis usisObj = new Usis();  System.out.println("2\*\*\*\*\*\*\*\*\*\*\*\*\*");  usisObj.login(rakib);  System.out.println("3\*\*\*\*\*\*\*\*\*\*\*\*\*");  usisObj.advising(rakib);  System.out.println("4\*\*\*\*\*\*\*\*\*\*\*\*\*");  rakib.email = "rakib@hotmail.com";  rakib.password = "1234";  System.out.println("5\*\*\*\*\*\*\*\*\*\*\*\*\*");  usisObj.login(rakib);  System.out.println("6\*\*\*\*\*\*\*\*\*\*\*\*\*");  usisObj.advising(rakib);  System.out.println("7\*\*\*\*\*\*\*\*\*\*\*\*\*");  usisObj.advising(rakib, "CSE110", "PHY111", "MAT110", "CSE260");  System.out.println("8\*\*\*\*\*\*\*\*\*\*\*\*\*");  usisObj.advising(rakib, "CSE110", "PHY111","MAT110");  System.out.println("9\*\*\*\*\*\*\*\*\*\*\*\*\*");  usisObj.allAdviseeInfo();  System.out.println("10\*\*\*\*\*\*\*\*\*\*\*\*\*");  roy.email = "roy@hotmail.com";  roy.password = "abcd";  usisObj.login(roy);  System.out.println("11\*\*\*\*\*\*\*\*\*\*\*\*\*");  usisObj.advising(roy, "CSE110", "ENG101", "PHY112");  System.out.println("12\*\*\*\*\*\*\*\*\*\*\*\*\*");  usisObj.allAdviseeInfo();  }  } | Student object is created  Student object is created  1\*\*\*\*\*\*\*\*\*\*\*\*\*  Usis is ready to use!  2\*\*\*\*\*\*\*\*\*\*\*\*\*  Email and password need to be set.  3\*\*\*\*\*\*\*\*\*\*\*\*\*  Please login to advise courses!  4\*\*\*\*\*\*\*\*\*\*\*\*\*  5\*\*\*\*\*\*\*\*\*\*\*\*\*  Login successful  6\*\*\*\*\*\*\*\*\*\*\*\*\*  You haven't selected any courses.  7\*\*\*\*\*\*\*\*\*\*\*\*\*  You need special approval to take more than 3 courses.  8\*\*\*\*\*\*\*\*\*\*\*\*\*  Advising successful!  9\*\*\*\*\*\*\*\*\*\*\*\*\*  Total Advisee: 1  Name: Rakib ID: 12301455  Department: CSE  Advised Courses:  CSE110 PHY111 MAT110  ==============  10\*\*\*\*\*\*\*\*\*\*\*\*\*  Login successful  11\*\*\*\*\*\*\*\*\*\*\*\*\*  Advising successful!  12\*\*\*\*\*\*\*\*\*\*\*\*\*  Total Advisee: 2  Name: Rakib ID: 12301455  Department: CSE  Advised Courses:  CSE110 PHY111 MAT110  ==============  Name: Roy ID: 12501345  Department: CS  Advised Courses:  CSE110 ENG101 PHY112  ============== |

# Task 9

|  |  |
| --- | --- |
| 1 | public class B{ |
| 2 | public int x = 3, y = 5, temp = -5, sum = 2; |
| 3 | public B(){ |
| 4 | y = temp + 3 ; |
| 5 | sum = 3 + temp + 2; |
| 6 | temp -= 2; |
| 7 | } |
| 8 | public B(B b){ |
| 9 | sum = b.sum; |
| 10 | x = b.x + 2; |
| 11 | b.methodB(2,3); |
| 12 | } |
| 13 | public void methodA(int m, int n){ |
| 14 | int x = 2; |
| 15 | y = y + m + (temp++); |
| 16 | x = x + 5 + n; |
| 17 | sum = sum + x + y; |
| 18 | System.out.println(x + " " + y+ " " + sum); |
| 19 | } |
| 20 | public void methodB(int m, int n){ |
| 21 | int y = 0; |
| 22 | y = y + this.y; |
| 23 | x = this.y + 2 + temp; |
| 24 | methodA(x, y); |
| 25 | sum = x + y + sum; |
| 26 | System.out.println(x + " " + y+ " " + sum); |
| 27 | } |
| 28 | } |

| public class Tester9 {  public static void main(String args []){  B b1 = new B();  B b2 = new B(b1);  b1.methodA(1, 2);  b2.methodB(3, 2);  }  } | **Outputs** | | |
| --- | --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Task 10

|  |  |
| --- | --- |
| 1 | public class msgClass{ |
| 2 | public int content; |
| 3 | } |
| 4 | class FinalT5A{ |
| 5 | public int sum = 2, y = 1, x = 1; |
| 6 | public void methodA(){ |
| 7 | int x=6, y =0; |
| 8 | msgClass myMsg = new msgClass(); |
| 9 | myMsg.content = this.x; |
| 10 | x = x + myMsg.content; |
| 11 | this.y = this.y + methodB(myMsg, myMsg.content); |
| 12 | System.out.println(x + " " + this.y+ " " + sum); |
| 13 | y = this.y/2 + this.x; |
| 14 | x = y + sum/2; |
| 15 | sum = x + y + myMsg.content; |
| 16 | System.out.println(x + " " + y+ " " + sum); |
| 17 | } |
| 18 | public int methodB(msgClass mg2, int mg1){ |
| 19 | int x = 0; |
| 20 | y = y + mg2.content; |
| 21 | mg2.content = y + mg1; |
| 22 | x = this.x + 3 + mg1; |
| 23 | sum = sum + x + y; |
| 24 | System.out.println(this.x + " " + this.y+ " " + sum); |
| 25 | mg2.content = sum - mg1 ; |
| 26 | return sum; |
| 27 | } |
| 28 | } |

|  |  |  |  |
| --- | --- | --- | --- |
| **DRIVER CODE** | **OUTPUTS** | | |
| public class Tester10{  public static void main(String args []){  FinalT5A fT5A = new FinalT5A();  fT5A.methodA();  }  } |  |  |  |
|  |  |  |
|  |  |  |

# 

# 

# Task 11

|  |  |
| --- | --- |
| **1** | **public class A{** |
| **2** | **public int temp = 3, sum = 9, y = 4, x = 0;** |
| **3** | **public A(){** |
| **4** | **int sum = 7;** |
| **5** | **y = temp - 5;** |
| **6** | **sum = temp + 2;** |
| **7** | **temp-=2;** |
| **8** | **this.x = sum + temp + y;** |
| **9** | **}** |
| **10** | **public A(int y, int temp){** |
| **11** | **y = temp - 1 + x;** |
| **12** | **sum = temp + 2 -x;** |
| **13** | **temp-=2;** |
| **14** | **}** |
| **15** | **public void methodA(int m, int [] n){** |
| **16** | **int x = 0;** |
| **17** | **y = y + m + methodB(x,m);** |
| **18** | **x = this.x + 2 + (++n[0]);** |
| **19** | **sum = sum + x + y;** |
| **20** | **n[0] = sum + 2;** |
| **21** | **System.out.println(n[0] + " " + y+ " " + sum);** |
| **22** | **}** |
| **23** | **public int methodB(int m, int n){** |
| **24** | **int [] y = {0};** |
| **25** | **this.y = y[0] + this.y + m;** |
| **26** | **x = this.y + 2 + temp - n;** |
| **27** | **sum = x + y[0] + this.sum;** |
| **28** | **System.out.println(y[0]+ " "+ temp + " " + sum);** |
| **29** | **return y[0];** |
| **30** | **}** |
| **31** | **}** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Driver Code** | **Output** | | |
| public class Tester11 {  public static void main(String args[]){  int[] x = {35};  A a1 = new A();  A a2 = new A(-5,-7);  a1.methodA(1, x);  a2.methodA(1, x);  }  } |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

#### 

# 

# Ungraded Tasks (Optional)

(You don’t have to submit the ungraded tasks)

## Task 1

Design the **Parcel** class in such a way that it produces the following output.

**NOTE:** For the method ***calcFee()***, if the delivery location is ***Dhanmondi***, then the location charge will be 50 taka or else it’ll be free. Also, while calculating total fee, if the product weight is 0 the total\_fee would also be 0.

**Formula:** fee = (weight \* 20) + *location\_charge (if any)*

|  |  |
| --- | --- |
| **Driver Code** | **Expected Output** |
| public class ParcelDriver {  public static void main(String[] args){  Parcel p1 = new Parcel();  p1.printDetails();  p1.name = "Spongebob";  p1.printDetails();  System.out.println("1\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  Parcel p2 = new Parcel("Bob the Builder");  p2.weight = 15;  p2.calcFee("Gulshan");  p2.printDetails();  System.out.println("2\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  p2.addWeight(25);  p2.calcFee("Banani");  p2.printDetails();  System.out.println("3\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  Parcel p3 = new Parcel("Dora the Explorer", 10);  p3.addWeight(15);  p3.calcFee("Dhanmondi");  p3.printDetails();  }  } | Set name first  Name: Spongebob  Total Weight: 0  Total Fee: 0.0  1\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Name: Bob the Builder  Total Weight: 15  Total Fee: 300.0  2\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Updated Weight: 40  Name: Bob the Builder  Total Weight: 40  Total Fee: 800.0  3\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Updated Weight: 25  Name: Dora the Explorer  Total Weight: 25  Total Fee: 550.0 |

# Task 2

Design the program to get the output as shown.

**Hints:**

* Create an array in the Team class to store the player’s object
* Use constructor overloading technique for Team class

|  |  |
| --- | --- |
| public class TeamTester {  public static void main(String[] args) {  Team b = new Team();  b.updateName("Bangladesh");  Player mashrafi = new Player("Mashrafi", 42, 100);  b.addPlayer(mashrafi);  Player tamim = new Player("Tamim", 35, 70);  b.addPlayer(tamim);  b.printDetail();  System.out.println("==============");  Team a = new Team("Australia");  Player ponting = new Player("Ponting", 50, 300);  a.addPlayer(ponting);  Player lee = new Player("Lee", 49, 200);  a.addPlayer(lee);  a.printDetail();  }  } | **Output:**  Team: Bangladesh  List of players:  Name: Mashrafi  Age: 42, Total Matches: 100  Name: Tamim  Age: 35, Total Matches: 70  ==============  Team: Australia  List of players:  Name: Ponting  Age: 50, Total Matches: 300  Name: Lee  Age: 49, Total Matches: 200 |

## Task 3

|  |  |
| --- | --- |
| 1 | public class TracingX { |
| 2 | public int x, y = 1; |
| 3 | public int metA(int y){ |
| 4 | y += x + 3; |
| 5 | int temp = y + this.y; |
| 6 | if (temp % 2 == 0){ |
| 7 | return temp; |
| 8 | } |
| 9 | TracingX t = new TracingX(); |
| 10 | t.y = this.x - (++x) + t.x; |
| 11 | this.y = y + t.metA(t.x); |
| 12 | System.out.println(x +" "+ y +" "+temp); |
| 13 | return temp+this.y; |
| 14 | } |
| 15 | } |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Driver code:  public class TesterX {  public static void main(String[] args) {  TracingX t1 = new TracingX();  t1.y = t1.x = 5;  TracingX t2 = new TracingX();  t2.x = t1.metA(2);  t2.y = t2.metA(4);  System.out.println(t1.y +t1.x +" "+t2.x +" "+t2.y);  }  } | Output:   |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  | |

# 

# 