| **1** | **public class Trace1 {** |
| --- | --- |
| **2** | **public int p = 3, y = 2, sum;** |
| **3** | **public void methodA(){** |
| **4** | **int x = 0, y = 0;** |
| **5** | **y = y + this.y;** |
| **6** | **x = sum + 2 + p;** |
| **7** | **sum = x + methodB(p, y) + this.y;** |
| **8** | **System.*out*.println(x + " " + y+ " " + sum);** |
| **9** | **}** |
| **10** | **public int methodB(int p, int n){** |
| **11** | **int x = 0;** |
| **12** | **y = this.y + (++p);** |
| **13** | **x = x + 2 + n;** |
| **14** | **sum = sum + x + y;** |
| **15** | **System.*out*.println(x + " " + y+ " " + sum);** |
| **16** | **return sum;** |
| **17** | **}** |
| **18** | **}** |

**Driver code:**

| public class Tester1 {  public static void main(String [] args){  Trace1 t1 = new Trace1();  t1.methodA();  t1.methodA();  Trace1 t2 = new Trace1();  System.out.println(t2.methodB(2,3));  }  } | **Outputs** | | |
| --- | --- | --- | --- |
| **x** | **y** | **Sum** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  | | |

Write the code in java for the **“Vehicle”** class. The tester class and the output is given below:

| **Tester class** | **Output** |
| --- | --- |
| public class TesterVehicle{  public static void main(String [] args){  Vehicle car = new Vehicle();  System.out.println("Attributes of car object:");  System.out.println(car.type);  System.out.println(car.wheels);  System.out.println(car.color);  System.out.println("=========");  Vehicle bike = new Vehicle();  bike.type="Motor bike";  bike.wheels=2;  bike.color="Red";  System.out.println("Attributes of bike object:");  System.out.println(bike.type);  System.out.println(bike.wheels);  System.out.println(bike.color);  }  } | Attributes of car object:  Car  4  White  =========  Attributes of bike object:  Motor bike  2  Red |

| **Driver Code** | **Output** |
| --- | --- |
| **public class TesterCourse{**  **public static void main(String[] args) {**  **Course c1 = new Course();**  **Course c2 = new Course();**  **System.*out*.println("========== 1 ==========");**  **c1.createCourse("Programming Language I", "CSE110", 3);**  **c1.displayCourse();**  **System.*out*.println("========== 2 ==========");**  **c2.createCourse("Data Structures", "CSE220", 3);**  **c2.displayCourse();**  **System.*out*.println("========== 3 ==========");**  **c1.updateCourse("Programming Language II", "CSE111", 3);**  **c1.displayCourse();**  **}**  **}** | **========== 1 ==========**  **Course Name: Programming Language I**  **Course Code: CSE110**  **Course Credit: 3**  **========== 2 ==========**  **Course Name: Data Structures**  **Course Code: CSE220**  **Course Credit: 3**  **========== 3 ==========**  **Course Name: Programming Language II**  **Course Code: CSE111**  **Course Credit: 3** |

Create a **Dog** class so that the tester code generates the given output:

| **Driver Code** | **Expected Output** |
| --- | --- |
| **public class TesterDog{**  **public static void main (String[] args) {**  **Dog scooby = new Dog();**  **Dog oldie = new Dog();**  **Dog goofy = new Dog();**    **scooby.changeName("Scooby");**  **goofy.changeName("Goofy");**    **System.*out*.println("1. ===============");**  **System.*out*.println(scooby.bark());**  **System.*out*.println("2. ===============");**  **System.*out*.println(oldie.bark());**  **System.*out*.println("3. ===============");**  **oldie.changeColor("White");**  **System.*out*.println("4. ===============");**  **System.*out*.println(oldie.bark());**  **System.*out*.println("5. ===============");**  **System.*out*.println(goofy.bark());**  **System.*out*.println("6. ===============");**  **scooby.changeColor("Brown");**  **System.*out*.println("7. ===============");**  **System.*out*.println(scooby.bark());**  **System.*out*.println("8. ===============");**  **goofy.changeColor("Black");**  **}**  **}** | **1. ===============**  **Scooby is barking**  **2. ===============**  **A dog is barking**  **3. ===============**  **This dog is White**  **4. ===============**  **White dog is barking**  **5. ===============**  **Goofy is barking**  **6. ===============**  **Scooby is Brown**  **7. ===============**  **Scooby the Brown dog is barking**  **8. ===============**  **Goofy is Black** |

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

* A reader will have a name, capacity to read and an array of books they are reading.
* The initial capacity of a reader will be 0. The initial name will be “New user”.
* A new array is created every time a reader’s capacity is increased, which replaces the initial array.

| **Driver Code** | **Expected Output** |
| --- | --- |
| **public class Reader\_tester {**  **public static void main(String[] args){**  **Reader r1 = new Reader();**  **Reader r2 = new Reader();**    **r1.createReader("Albert", 2);**  **r2.createReader("Issac", 5);**    **System.out.println("1 ==========");**  **r1.readerInfo();**    **System.out.println("2 ==========");**  **r2.addBook("Java");**  **r2.addBook("Python");**  **r2.addBook("C++");**  **r2.readerInfo();**    **System.out.println("3 ==========");**  **r1.addBook("C#");**  **r1.addBook("Rust");**  **r1.addBook("GoLang");**    **System.out.println("4 ==========");**  **r1.increaseCapacity(5);**  **r1.addBook("Python");**    **System.out.println("5 ==========");**  **r1.readerInfo();**  **}**  **}** | **1 ==========**  **Name: Albert**  **Capacity: 2**  **Books:**  **No books added yet**  **2 ==========**  **Name: Issac**  **Capacity: 5**  **Books:**  **Book 1: Java**  **Book 2: Python**  **Book 3: C++**  **3 ==========**  **No more space for new book**  **4 ==========**  **Albert's capacity increased to 5**  **5 ==========**  **Name: Albert**  **Capacity: 5**  **Books:**  **Book 1: C#**  **Book 2: Rust**  **Book 3: Python** |

| **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** | **this.x = sum + --temp + y;** |
| **8** | **}** |
| **9** | **public A(int y, int temp){** |
| **10** | **y = temp - 1 + x;** |
| **11** | **sum = temp + 2 - x;** |
| **12** | **temp -= 2;** |
| **13** | **}** |
| **14** | **public void methodA(int m, int n){** |
| **15** | **int x = 0;** |
| **16** | **y = y + methodB(x,m) + m;** |
| **17** | **x = this.x + 2 + ++n;** |
| **18** | **sum = sum + y + methodB(n);** |
| **19** | **System.out.println(x + " " + y+ " " + sum);** |
| **20** | **}** |
| **21** | **public int methodB(int m, int n){** |
| **22** | **int y = 0;** |
| **23** | **this.y = y + this.y + m;** |
| **24** | **x = this.y + 2 + temp - n;** |
| **25** | **sum = x + y + this.sum;** |
| **26** | **System.out.println(y+ " "+ temp + " " + sum);** |
| **27** | **return y;** |
| **28** | **}** |
| **29** | **public int methodB(int x){** |
| **30** | **this.x = x + this.y;** |
| **31** | **x -= y + 2 + temp;** |
| **32** | **sum = x + y + this.sum;** |
| **33** | **return sum;** |
| **34** | **}** |
| **35** | **}** |

| **Driver Code** | **Output** | | |
| --- | --- | --- | --- |
| public class Tester11 {  public static void main(String args[]){  A a1 = new A();  A a2 = new A(-5,-7);  a1.methodA(1, 2);  a2.methodA(1, 4);  }  } |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Design the **Course** class with the necessary properties so that the given output is produced for the provided driver code.

| **Driver Class** | **Output** |
| --- | --- |
| public class CourseTester{  public static void main(String [] args){  Course c1 = new Course("PL II", "CS11");  System.out.println("--------1--------");  System.out.println(c1.printDetails());  System.out.println("--------2--------");  c1.addContent("Overloading");  System.out.println(c1.printDetails());  System.out.println("--------3--------");  c1.addContent("Encapsulation");  c1.addContent("Static", "Polymorphism");  System.out.println(c1.printDetails());  System.out.println("--------4--------");  c1.addContent("Inheritance");  System.out.println("--------5--------");  Course c2 = new Course("DS", "CS22");  c2.addContent("Stack");  c2.addContent("Recursion","Tree");  c2.addContent("Heap","Hashing");  System.out.println("--------6--------");  System.out.println(c2.printDetails());  System.out.println("--------7--------");  Course c3 = new Course("OS");  c3.addContent("Scheduling");  System.out.println("--------8--------");  c3.addContent(new String[]{"Segmentation", "Process", "Interrupt","concurrency"});  System.out.println("--------9--------");  System.out.println(c3.printDetails());  }  } | --------1--------  Course details:  Course Name: PL II  Course Code: CS11  Course Syllabus:  No content yet.  --------2--------  Overloading was added.  Course details:  Course Name: PL II  Course Code: CS11  Course Syllabus:  Overloading  --------3--------  Encapsulation was added.  Static was added.  Polymorphism was added.  Course details:  Course Name: PL II  Course Code: CS11  Course Syllabus:  Overloading, Encapsulation, Static, Polymorphism  --------4--------  Cannot add more content  --------5--------  Stack was added.  Recursion was added.  Tree was added.  Heap was added.  Cannot add more content  --------6--------  Course details:  Course Name: DS  Course Code: CS22  Course Syllabus:  Stack, Recursion, Tree, Heap  --------7--------  Scheduling was added  --------8--------  Segmentation was added  Process was added  Interrupt was added  Cannot add more content  --------9--------  Course details:  Course Name: OS  Course Code: Undefined  Course Syllabus:  Scheduling, Segmentation, Process, Interrupt |

## **Reference Passing**

Constructor with multiclass

A team can have a maximum of 4 players

| Driver | Output |
| --- | --- |
| **public class TeamTester {**  **public static void main(String[] args) {**  **Player p1 = new Player("Sakep", 37, 200);**  **Player p2 = new Player("Tamem", 35, 180);**  **p1.details();**  **System.*out*.println("=======1=======");**  **p2.details();**  **System.*out*.println("=======2=======");**  **p1.addRival(p2);**  **System.*out*.println("=======3=======");**  **p1.showRival();**  **System.*out*.println("=======4=======");**  **Team b = new Team();**  **b.updateName("Bangladesh");**  **System.*out*.println("=======5=======");**  **b.addPlayer(p1);**  **System.*out*.println("=======6=======");**  **b.addPlayer(p2);**  **System.*out*.println("=======7=======");**  **b.printDetails();**  **System.*out*.println("=======8=======");**  **Team a = new Team("Australia");**  **Player p3 = new Player("Ponting", 50, 300);**  **a.addPlayer(p3);**  **Player p4 = new Player("Lee", 49, 200);**  **a.addPlayer(p4);**  **a.printDetails();**  **}**  **}** | **Name: Sakep**  **Age: 37, Total matches: 200**  **=======1=======**  **Name: Tamem**  **Age: 35, Total matches: 180**  **=======2=======**  **=======3=======**  **Rival of Sakep is Tamem**  **Rival’s stats:**  **Name: Tamem**  **Age: 35, Total matches: 180**  **=======4=======**  **=======5=======**  **Player Sakep added**  **=======6=======**  **Player Tamem added**  **=======7=======**  **Team: Bangladesh**  **List of players:**  **Name: Sakep**  **Age: 37, Total Matches: 200**  **Name: Tamem**  **Age: 35, Total Matches: 180**  **=======8=======**  **Player Ponting added**  **Player Lee added**  **Team: Australia**  **List of players:**  **Name: Ponting**  **Age: 50, Total Matches: 300**  **Name: Lee**  **Age: 49, Total Matches: 200** |

| **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** | **sum += x;** |
| **13** | **}** |
| **14** | **public void methodA(int m, int n){** |
| **15** | **int x = 2;** |
| **16** | **y = y + m + (temp++);** |
| **17** | **x = x + 5 + n;** |
| **18** | **sum = sum + x + y;** |
| **19** | **System.out.println(x + " " + y+ " " + sum);** |
| **20** | **}** |
| **21** | **public void methodB(int m, int n){** |
| **22** | **int y = 0;** |
| **23** | **y = y + this.y;** |
| **24** | **x = this.y + 2 + temp;** |
| **25** | **methodA(x, y);** |
| **26** | **sum = x + y + sum;** |
| **27** | **System.out.println(x + " " + y+ " " + sum);** |
| **28** | **}** |
| **29** | **}** |

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

| **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:   |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

**Design the Club class:**

| **Driver** | **Output** |
| --- | --- |
| public class ClubTester {  public static void main(String[] args) {  Club club1 = new Club();  System.*out*.println("1=================");  System.*out*.println(club1.approveClub("Makers Club",4,10000, 3));  System.*out*.println("2=================");  System.*out*.println(club1.approveClub("Makers Club",10,10000, 3));  System.*out*.println("3=================");  club1.info();  System.*out*.println("4=================");  club1.createEvent("Exhibit", 1099, 5);  System.*out*.println("5=================");  club1.createEvent("Impromptu", 8700, 6);  System.*out*.println("6=================");  club1.recruitMember(5);  System.*out*.println("7=================");  club1.createEvent("Impromptu", 8700, 6);  System.*out*.println("8=================");  club1.info();  System.*out*.println("9=================");  club1.createEvent("Potluck", 1200, 3);  System.*out*.println("10=================");  club1.createEvent("Potluck", 100, 3);  System.*out*.println("11=================");  club1.info();  System.*out*.println("12=================");  club1.createEvent("Speech", 100, 2);  System.*out*.println("13=================");  club1.endEvent("Exhibit");  System.*out*.println("14=================");  club1.info();  System.*out*.println("15=================");  club1.createEvent("Speech",100, 2);  }  } | 1=================  A club must have at least 5 members  2=================  New club, Makers Club, created with 10 members.  3=================  Name of club: Makers Club  Non-working members: 10  Current Budget: 10000.0  No events yet.  4=================  New event, "Exhibit" has started!  5 members are now working.  5=================  Need 1 more member(s) to arrange.  6=================  New members recruited  Total non-working members now are 10.  7=================  New event, "Impromptu" has started!  6 available members are now working.  8=================  Name of club: Makers Club  Non-working members: 4  Current Budget: 201.0  Events:  Exhibit  Impromptu  9=================  Not enough budget.  10=================  New event, "Potluck" has started!  3 members are now working.  11=================  Name of club: Makers Club  Non-working members: 1  Current Budget: 101.0  Events:  Exhibit  Impromptu  Potluck  12=================  Need 1 more member(s) to arrange.  13=================  Exhibit has ended!  5 members are free now.  14=================  Name of club: Makers Club  Non-working members: 6  Current Budget: 101.0  Events:  Impromptu  Potluck  15=================  New event, "Speech" has started!  2 members are now working. |

| **Solution:**  public class Club {  public String name;  public int member;  public double budget;  public int events;  public String eventName[];  public int eventMem[];  public String approveClub(String n, int m, double budget, int cap) {  if (m >= 5) {  name = n;  member = m;  this.budget = budget;  events = cap;  eventName = new String[cap];  eventMem = new int[cap];  return "New club, " + name + ", created with " + member + " members.";  } else {  return "A club must have at least 5 members";  }  }  public void info() {  System.*out*.println("Name of the club: " + name);  System.*out*.println("Non-working Members: " + member);  System.*out*.println("Current Budget: " + budget);  if (events == eventName.length)  System.*out*.println("No events yet.");  else {  System.*out*.println("Current events: ");  for (int i = 0; i < eventName.length; i++) {  if (eventName[i] != null) {  System.*out*.println(eventName[i]);  }  }  }  }  public void createEvent(String ename, double ebud, int emem) {  if (budget < ebud) {  System.*out*.println("Not enough budget.");  } else if (member < emem) {  System.*out*.println("Need " + (emem - member) + " more member(s) to arrange.");  } else if (events == 0) {  System.*out*.println("Only " + eventName.length + " events can be created.");  } else {  System.*out*.println("New event, \"" + ename + "\"has started!\n" + emem + " out of " + member + " available members are now working.");  for (int i = 0; i < eventName.length; i++) {  if (eventName[i] == null) {  eventName[i] = ename;  eventMem[i] = emem;  member = member - emem;  budget = budget - ebud;  events--;  break;  }  }  }  }  public void recruitMember(int m) {  member += m;  System.*out*.println("New members recruited\n" +  "Total non-working members now are " + member);  }  public void endEvent(String ename) {  for (int i = 0; i < eventName.length; i++) {  if (eventName[i] != null && eventName[i].equals(ename)) {  eventName[i] = null;  member = member + eventMem[i];  eventMem[i] = 0;  events++;  break;  }  }  System.*out*.println(ename + " has ended!\n" +  member + " members are free now.");  }  } |
| --- |

## **Multi Class**

| **1** | **public class examClass {** |
| --- | --- |
| **2** | **public int ques;** |
| **3** | **public int sum;** |
| **4** | **public void methodA() {** |
| **5** | **System.out.println(ques + " " + 0 + " " + 0);** |
| **6** | **}** |
| **7** | **}** |
| **8** | **class QuizA {** |
| **9** | **public int x, y;** |
| **10** | **public int sum = 1;** |
| **11** | **public QuizA(int x, int y) {** |
| **12** | **this.x = y;** |
| **13** | **this.y = x;** |
| **14** | **}** |
| **15** | **public void methodA() {** |
| **16** | **int x = 3;** |
| **17** | **y = this.y + x;** |
| **18** | **examClass exam = new examClass();** |
| **19** | **exam.sum = x;** |
| **20** | **exam.ques = this.y;** |
| **21** | **x = this.x + x + exam.sum;** |
| **22** | **this.y = this.sum + methodB(exam.ques, exam);** |
| **23** | **System.out.println(x + " " + this.y + " " + sum);** |
| **24** | **sum = x % 2 + this.x;** |
| **25** | **y = x + y + exam.sum;** |
| **26** | **System.out.println(x + " " + y + " " + sum);** |
| **27** | **}** |
| **28** | **public int methodB(int x1, examClass x2) {** |
| **29** | **int y = 0;** |
| **30** | **y = this.y + x2.sum;** |
| **31** | **x2.ques = x1 + x2.ques;** |
| **32** | **sum = sum + x + y;** |
| **33** | **System.out.println(this.x + " " + this.y + " " + sum);** |
| **34** | **return x2.sum;** |
| **35** | **}** |
| **36** | **}** |

| **Driver Code** | **Output** | | |
| --- | --- | --- | --- |
| **public class QuizTesterA{**  **public static void main(String []args){**  **QuizA q1 = new QuizA(3,4);**  **q1.methodA();**  **}**  **}** |  |  |  |
|  |  |  |
|  |  |  |

## **Encapsulation + Multi Class**

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.
3. All the variables of BracuStudent class must be private

| **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.givePass();  st3.givePass();  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.givePass();  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 |

## **Static**

Design the Event class based on the following Tester Class and Output.

* The instance & static variables of the Class Must Be Private.
* The instance & static methods of the Class Must Be Public.
* At most only 3 events can be booked.

| **Tester Class** | **Expected Output** |
| --- | --- |
| public class EventTester {  public static void main(String[] args){  Event.showEvents();  System.out.println("==========1==========");  Event p1 = new Event();  Event.addEvent( p1 );  System.out.println("------------");  Event.showEvents();  System.out.println("==========2==========");  p1.setInfo("Abeda", 12);  System.out.println("------------");  Event.addEvent( p1 );  Event p2 = new Event();  p2.setInfo("Mashrur", 25);  Event.addEvent( p2 );  System.out.println("==========3==========");  p2.setHeadCount(9);  Event.addEvent( p2 );  System.out.println("==========4==========");  Event p3 = new Event();  p3.setInfo("Sanjeet", 3);  System.out.println(p3.getEventInfo());  Event.addEvent(p3);  System.out.println("==========5==========");  Event.showEvents();  System.out.println("==========6==========");  Event p4 = new Event();  p4.setInfo("Ayesha",2 );  Event.addEvent(p4);  }  } | No Events Yet.  Capacity Left: 30  ==========1==========  Event booking Failed! Info missing.  ------------  No Events Yet.  Capacity Left: 30  ==========2==========  Abeda's headcount is 12 people  ------------  Abeda's Event has been booked!!  Mashrur's headcount is 25 people  Event booking Failed! Not enough Capacity.  ==========3==========  Mashrur's Event has been booked!!  ==========4==========  Sanjeet's headcount is 3 people  Organizer: Sanjeet, HeadCount: 3  Sanjeet's Event has been booked!!  ==========5==========  Event#1  Event By: Abeda, HeadCount: 12  Event#2  Event By: Mashrur, HeadCount: 9  Event#3  Event By: Sanjeet, HeadCount: 3  Capacity Left: 6  ==========6==========  Ayesha's headcount is 2 people  Event booking Failed! Max organizers reached. |

Write the **SultansDine** class so that the given code provides the expected output.

1. Create **SultansDine** class
2. Create 2 static variable and 1 static array
3. Create 1 static method
4. Calculation of branch sell is given below
   1. If sellQuantity < 10:
      1. Branch\_sell = quantity \* 300
   2. Else if sellQuantity < 20:
      1. Branch\_sell = quantity \* 350
   3. Else
      1. Branch\_sell = quantity \* 400
5. Calculation of branch’s sell percentage = (branch’s sell / total sell) \* 100

| ***# Write your code here***  public class SDTester{  public static void main(String [] args){  SultansDine.details();  System.out.println("########################");  SultansDine dhanmondi = new SultansDine("Dhanmondi");  dhanmondi.sellQuantity(25);  dhanmondi.branchInformation();  System.out.println("-----------------------------------------");  SultansDine.details();  System.out.println("========================");  SultansDine baily\_road = new SultansDine("Baily Road");  baily\_road.sellQuantity(15);  baily\_road.branchInformation();  System.out.println("-----------------------------------------");  SultansDine.details();  System.out.println("========================");  SultansDine gulshan = new SultansDine("Gulshan");  gulshan.sellQuantity(9);  gulshan.branchInformation();  System.out.println("-----------------------------------------");  SultansDine.details();  }  } | ***Output:***  Total Number of branch(s): 0  Total Sell: 0 Taka  #################################  Branch Name: Dhanmondi  Branch Sell: 10000 Taka  -----------------------------------------  Total Number of branch(s): 1  Total Sell: 10000 Taka  Branch Name: Dhanmondi, Branch Sell: 10000 Taka  Branch consists of total sell's: 100.00%  ================================  Branch Name: Baily Road  Branch Sell: 5250 Taka  -----------------------------------------  Total Number of branch(s): 2  Total Sell: 15250 Taka  Branch Name: Dhanmondi, Branch Sell: 10000 Taka  Branch consists of total sell's: 65.57%  Branch Name: Baily Road, Branch Sell: 5250 Taka  Branch consists of total sell's: 34.43%  ================================  Branch Name: Gulshan  Branch Sell: 2700 Taka  -----------------------------------------  Total Number of branch(s): 3  Total Sell: 17950 Taka  Branch Name: Dhanmondi, Branch Sell: 10000 Taka  Branch consists of total sell's: 55.71%  Branch Name: Baily Road, Branch Sell: 5250 Taka  Branch consists of total sell's: 29.25%  Branch Name: Gulshan, Branch Sell: 2700 Taka  Branch consists of total sell's: 15.04% |
| --- | --- |

# 

# 

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

**Consider the following code:**

| B b1 = new B();  B b2 = new B(b1);  b1.methodA(3, 2);  b2.methodB(1, 2)**;** | **x** | **y** | **sum** |
| --- | --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

| 1 | public class TracingExample { |
| --- | --- |
| 2 | public static double counter = 3.0; |
| 3 | public static int a, b; |
| 4 | public TracingExample(int a, int b) { |
| 5 | this.a = a; |
| 6 | b = (a + b) % 2; |
| 7 | counter++; |
| 8 | } |
| 9 | public void methodA(int value) { |
| 10 | this.a += value; |
| 11 | int z = (a + b) % 3; |
| 12 | System.out.println(this.a + " " + z + counter); |
| 13 | if (a >= 15) { |
| 14 | this.methodB(this); |
| 15 | System.out.println(z + counter + " " + this.a); |
| 16 | } |
| 17 | } |
| 18 | public static void methodA(double value) { |
| 19 | a += value; |
| 20 | int z = (a + b) % 3; |
| 21 | System.out.println(a + " " + 1 + z + counter); |
| 22 | a += 2; |
| 23 | } |
| 24 | public static void methodB(TracingExample obj) { |
| 25 | double temp = (counter \* 2) + 2; |
| 26 | obj.a += temp; |
| 27 | System.out.println(obj.a + " " + temp); |
| 28 | } |
| 29 | } |

| public class TestTracing {  public static void main(String[ ] args) {  TracingExample t1 = new TracingExample(2, 3);  TracingExample t2 = new TracingExample(5, 7);  t1.methodA(TracingExample.counter);  t1.methodA(7);  TracingExample.methodA(3.0);  }  } |
| --- |

| 1 | class Task{ |
| --- | --- |
| 2 | public static int x; |
| 3 | public int y; |
| 4 | public Task(){ |
| 5 | x = this.x + 5; |
| 6 | } |
| 7 | public Task(int x){ |
| 8 | this(); |
| 9 | this.x = x + 10; |
| 10 | } |
| 11 | public static void met(Task [] t){ |
| 12 | int x = t[0].met(t[1]) + t[1].x++; |
| 13 | int y = ++x + t[1].met(t[0]); |
| 14 | System.out.println(x + " " + y); |
| 15 | } |
| 16 | public int met(Task t){ |
| 17 | int x = (int)(t.x % 2 - y + 2); |
| 18 | t.y = x + this.y; |
| 19 | System.out.println(x + " " +t.y); |
| 20 | return x; |
| 21 | } |
| 22 | } |
| 23 |  |
| 24 | public class Tester{ |
| 25 | public static void met(Task[]t){ |
| 26 | Task.met(t); |
| 27 | } |
| 28 | public static void main(String[] args) { |
| 29 | Task t1 = new Task(1); |
| 30 | Task t2 = new Task(2); |
| 31 | **Task [] ta = {t1, t2};** |
| 32 | met(ta); |
| 33 | } |
| 34 | } |

## **Inheritance**

| **1** | **public class A {** |
| --- | --- |
| **2** | **static int temp = 4;** |
| **3** | **static int x = -10;** |
| **4** | **int sum, y;** |
| **5** | **public A() {** |
| **6** | **y = temp - 2;** |
| **7** | **sum = temp + 1;** |
| **8** | **temp -= 2;** |
| **9** | **}** |
| **10** | **public void methodA(int m, int n) {** |
| **11** | **y = y + m + (temp++);** |
| **12** | **x = x + 1 + n;** |
| **13** | **sum = sum + x + y;** |
| **14** | **System.out.println(x + " " + y + " " + sum);** |
| **15** | **}** |
| **16** | **}** |
| **17** | **class B extends A {** |
| **18** | **static int x = 0;** |
| **19** | **int sum = -6;** |
| **20** | **public B() {** |
| **21** | **sum = 0;** |
| **22** | **y = temp + 3;** |
| **23** | **super.sum = 3 + temp + 2;** |
| **24** | **temp -= 2;** |
| **25** | **}** |
| **26** | **public B(B b) {** |
| **27** | **sum = b.sum + super.sum;** |
| **28** | **x = b.x + 1;** |
| **29** | **b.methodB(2, 3);** |
| **30** | **}** |
| **31** | **public void methodB(int m, int n) {** |
| **32** | **int y = 0;** |
| **33** | **y = y + this.y;** |
| **34** | **x = y + 2 + (++temp);** |
| **35** | **methodA(x, y);** |
| **36** | **sum = x + y + sum;** |
| **37** | **System.out.println(x + " " + y + " " + sum);** |
| **38** | **}** |
| **39** | **}** |

| **public class Tester {**  **public static void main(String [] args) {**  **A a1 = new A();**  **B b1 = new B();**  **B b2 = new B(b1);**  **b1.methodA(2, 3);**  **b2.methodB(3, 8);**  **}**  **}** | **Output:** | | |
| --- | --- | --- | --- |
| **x** | **y** | **sum** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Design the **Manager** and **Developer** class derived from the **Employee** class with appropriate attributes and properties so that the driver code can generate the output given below. [**Hint**:

Manager:

1. Adds a bonus to the base salary if the manager works more than 40 hours.
2. If the manager works more than 100 hours, the full amount is approved; if they work more than 80 hours, half the amount is approved. Otherwise, the increment is denied.

Developer:

1. Adds $700 to the base salary if the developer works with Java programming language.**]**

| **Driver Code and Parent Class** | **Output** |
| --- | --- |
| public class EmployeeTester {  public static void main(String[] args) {  Manager neymar = new Manager("Neymar",1000, 45, 10);  Developer messi = new Developer("Messi",1000,50,"Java");  Developer chiesa = new Developer("Chiesa", 1000, 50, "Javascript");  neymar.calculateSalary();  System.out.println("1.==========");  neymar.displayInfo();  System.out.println("2.==========");  neymar.requestIncrement(100);  System.out.println("3.==========");  neymar.setHoursWorked(85);  neymar.requestIncrement(100);  System.out.println("4.==========");  neymar.calculateSalary();  System.out.println("5.==========");  neymar.displayInfo();  System.out.println("6.==========");  messi.calculateSalary();  System.out.println("7.==========");  messi.displayInfo();  System.out.println("8.==========");  chiesa.calculateSalary();  System.out.println("9.==========");  chiesa.displayInfo();  } }  public class Employee {  public String name;  private double baseSalary;  private int hoursWorked;  public Employee(String name, double baseSalary, int hoursWorked){  this.name = name;  this.baseSalary = baseSalary;  this.hoursWorked = hoursWorked;  }  public double getBaseSalary() {  return baseSalary;  }  public void setBaseSalary(double baseSalary) {  this.baseSalary = baseSalary;  }  public int getHoursWorked() {  return hoursWorked;  }  public void setHoursWorked(int hoursWorked) {  this.hoursWorked = hoursWorked;  }  public void displayInfo() {  System.out.println("Name: " + name);  System.out.println("Base Salary: $" + baseSalary);  System.out.println("Work Hours: " + hoursWorked);  } } | 1.==========  Name: Neymar  Base Salary: $1000.0  Work Hours: 45  Bonus: 10.0 %  Final Salary: $1100.0  2.==========  Increment denied.  3.==========  $50 Increment approved.  4.==========  5.==========  Name: Neymar  Base Salary: $1050.0  Work Hours: 85  Bonus: 10.0 %  Final Salary: $1155.0  6.==========  7.==========  Name: Messi  Base Salary: $1000.0  Work Hours: 50  Language: Java  Final Salary: $1700.0  8.==========  Name: Chiesa  Base Salary: $1000.0  Work Hours: 50  Language: Javascript  Final Salary: $1000.0 |

## 