| Points:   1. Don’t use default package 2. Class name should not be Demo,Test,X,A,B 3. Variable names should not be a,b,c 4. Variable names should start with small latter 5. Every variable should be private except local 6. Method names should not be m1,m2 7. Method names should start with small latter |
| --- |

| public class Employee {  private int id;  private String name;  private int sal;  public Employee(int id, String name, int sal) {  super();  this.id = id;  this.name = name;  this.sal = sal;  }  public Employee() {    }  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public int getSal() {  return sal;  }  public void setSal(int sal) {  this.sal = sal;  }  }  **Notes:whenever I say Employee class, refer to this class.** |
| --- |
|  |

| **1.**Create Product class with fields id,name and price and create Product object and store data and display that data. |
| --- |
| **2.**public class Demo {  public static void main(String[] args) {  **//here create object of Employee class and add 101,sam,1000 data into that**  **//display that data here**  }  } |
| **3.** public class Demo {  public static void main(String[] args) {  **//here create object of Employee class and add 101,sam,1000 data into that using setter method**  **//display that data here**  }  } |
| **4.** public class Demo {  public static void main(String[] args) {  **//here create object of Employee class and add 101,sam,1000 data into that using constructor**  **//display that data here**  }  } |
| **5.**public class Demo {  public static void main(String[] args) {  **//here create object of Employee class and add 101,sam,1000 data into that**  **//call show method and pass this created object to show method**  }  public static void show(){  **//do required changes to show method and display that data here**  }  } |
| **6.**public class Demo {  public static void main(String[] args) {  **//here create object of Employee class and add 101,sam,1000 data into that**  **//call show method and pass this created object to show method**  }  public void show(){  **//do required changes to show method and display that data here**  }  } |
| **7.** public class Demo {  public static void main(String[] args) {  **//here create two objects of Employee class and add data into that**  **//call show method and pass this two objects to show method**  }  public static void show(){  **//do required changes to show method and display that data here**  }  } |
| **8. Find the output**  public class Demo {  public static void main(String[] args) {  Employee emp = new Employee(101, "sam", 1000);  Demo.show(emp);  System.out.println(emp.getId()+" " + emp.getName() + " "+ emp.getSal());  }  public static void show(Employee ex) {  System.out.println(ex.getId() + " " + ex.getName() + " " + ex.getSal());  ex.setId(102);  }  } |
| **9.**  public class Demo {  public static void main(String[] args) {  **//create Employee object with 101,sam,1000 data**  **//display this object data by passing to show method**  **//add 100 bonus in salary**  **//display this object data by passing to show method**  }  public static void show(){  **//do required changes in show method**  }  } |
| **10.**  public class Demo {  public static void main(String[] args) {  **//create Employee object with 101,sam,1000 data**  **//display this object data by passing to show method**  **//add 100 bonus in salary**  **//display this object data by passing to show method**  }  public void show(){  **//do required changes in show method**  }  } |
| **11. In below example ,try to call show method in different ways which explains different ways to call static methods**  public class Demo {  public static void main(String[] args) {  **//call show method in different ways**  }  public static void show(){  System.out.println("show method");  }  } |
| **12. In below example, try to print total variable in different ways which explains different ways to call static variable**  public class Demo {  private static int total = 10;  public static void main(String[] args) {  **//print total in different ways**  }  } |
| **13.** public class Demo {  int x = 10;  public void show(){  int x = 100;  System.out.println(x);**//do required changes at this line but output should be 100**  System.out.println(x);**//do required changes at this line but output should be 10**  }  public static void main(String[] args) {  Demo d = new Demo();  d.show();  }  } |
| **14.**  public class Test {    public void show(){  System.out.println("Hello world");  }  }  public class Demo {  public static void main(String[] args) {  **//call show method of Test class from here**  }  } |
| **15.**  public class Test {  public static void show(){  System.out.println("Hello world");  }  }  public class Demo {  public static void main(String[] args) {  **//call show method of Test class from here**  }  } |
| **16.**  public class Demo {  public static void m1(){  System.out.println(a);  }  public static void main(String[] args) {  **//call m1 method from here**  }  } |
| **17.**  public class Demo {  public void m1(){  System.out.println("m1 method");  }  public static void main(String[] args) {  //call m1 method from here  }  } |
| **18.**  public class Demo {  public static void m1(int a){  System.out.println(a);  }  public static void main(String[] args) {  **//call m1 method from here**  }  } |
| **19.**  public class Demo {  public void m1(float a){  System.out.println(a);  }  public static void main(String[] args) {  **//call m1 method from here**  }  } |
| **20. Find the output of following program**  public class Demo {  public static void m1(float a){  System.out.println(a);  }  public static void main(String[] args) {  m1(10);  }  } |
| **21.**  public class Demo {  public static void m1(Employee e){  System.out.println("m1 method");  }  public static void main(String[] args) {  **//call m1 method from here**  }  } |
| **22.**  public class Demo {  public static void m1(String s){  System.out.println("s");  }  public static void main(String[] args) {  **//call m1 method from here**  }  } |
| **23.**  public class Demo {  public static void m1(Object o){  System.out.println("m1 method");  }  public static void main(String[] args) {  **//call m1 method from here**  }  } |
| **24.**  public class Demo {  public static void m1(Employee e){  **//display data here**  }  public static void main(String[] args) {  **//call m1 method from here**  }  } |
| **25.**  public class Demo {  public static void m1(Object o){  **//display employee data here**  }  public static void main(String[] args) {  **//create Employee object here with 101,sam,1000 data**  **//call m1 method by passing this Employee object**  }  } |
| **26.**  public class Demo {  public static void m1(int i){  System.out.println(i);  }  public static void main(String[] args) {  **//create Employee object here with 101,sam,1000 data**  **//call m1 method and pass employee id as parameter**  }  } |
| **27.**  public class Demo {  public static int m1(int i){  System.out.println("m1 called");  return i\*i;  }  public static void main(String[] args) {  **//call m1 method and pass some number**  }  } |
| **28.**  public class Demo {  public static int m1(int i){  System.out.println("m1 called");  return i\*i;  }  public static void main(String[] args) {  **//call m1 method and pass some number**  int total = 100 + **use returned value here;**  System.out.println(total);  }  } |
| **29.**  public class Demo {  public static float m1(int i){  System.out.println("m1 called");  return i\*i;  }  public static void main(String[] args) {  **//call m1 method and pass some number**  int total = 100 + **use returned value here;**  System.out.println(total);  } |
| **30.**  public class Demo {  public static void m1(){  **//create Employee object here and return**  **//do required changes in m1 method to do this task**  }  public static void main(String[] args) {  **//call m1 method and catch returned employee object here**  **//display returned employee data here**  }  } |
| **31.**  public class Demo {  public static Object m1(){  **//create Employee object here and return**  **//don’t change m1 method prototype i.e return type or access specifier**  }  public static void main(String[] args) {  **//call m1 method and catch returned employee object here**  **//display returned employee data here**  }  } |
| **32. In below program, try to call m1 method in different ways like with catching returned value,without catching returned value,call from expression i.e different ways which will show how to call non void methods**  public class Demo {  public static int m1(int i){  return i\*i;  }  public static void main(String[] args) {  **//call m1 method from here in different ways**  }  } |
| **33.**  public class Demo {  public static void show(){  }  public static void process(){    }  public static void main(String[] args) {  **//do required changes in main,process and show method**  **//create empty employee object here**  **//add data into that empty object using process method**  **//display employee object with data in show method**  }  } |
| **34.**  public class Demo {  public static void main(String[] args) {  **//take data id,name,salary from user using scanner and display that data in show method**  }  public void show(){  **//display data here by doing required changes**  }  } |
| **35. Find the output**  public class Demo {  static int total = 0;  public static void add() {  int total = 10 + 30;  }  public static void main(String[] args) {  Demo.add();  System.out.println(total);  }  } |
| **36.Find the output**  public class Demo {  static int total = 0;  public static void add() {  total = 10 + 30;  }  public static void main(String[] args) {  Demo.add();  System.out.println(total);  } |
| **37. Write a program which reverse the string str**  public class Demo {  public static String reverseString(String str){  //reverse str string here and return  }  public static void main(String[] args) {  String str = "Hello Welcome";  String reversedString = reverseString(str);  System.out.println(reversedString);  }  } |
| **38.**  public class Demo {    public static void main(String[] args) {  String str = "Hello Welcome";  **//here print second character of str string for example e**  }  } |
| **39.**  public class Demo {    public static void main(String[] args) {  String str = "Hello how are you sam?";  **//here print the index of 'o' character from str string**  }  } |
| **40.**  public class Demo {    public static void main(String[] args) {  String str = "Hello how are you sam?";  **//here print the index of last 'o' character from str string**  }  } |
| **41.**  public class Demo {    public static void main(String[] args) {  String str = "Hello how are you shyam?";  **//here print the index of third occurrence 'o' character from str string**  }  } |
| **42.**  public class Demo {    public static void main(String[] args) {  String str = "Hello how are you shyam?";  **//here print number of occurrences of 'o' character from str string**  }  } |
| **43.**  public class Demo {  public static int calculateLength(String str){  **//calculate length here and don't use length function of string class**  }  public static void main(String[] args) {  String str = "Hello how are you shyam?";  int len = calculateLength(str);  System.out.println(len);  }  } |
| **44.**  public class Demo {  public static void main(String[] args) {  String str = "Hello Welcome";  **//print index of first 'l' after 'W'**  }  } |
| **45. Write a program which explains different access specifiers and it’s scope** |
| **46. Create tightly encapsulated Employee class** |
| **47. Create a class which explains you inheritance concept** |
| **48. Write a program which shows why java does not support multiple inheritance** |
| **49.**  class Test{    }  class Test1{    }  public class Demo{  public static void main(String[] args) {  **//create one method in class Test**  **//override that method in Test1 class**  **//call that overridden method from here**  **//to do this thing do required changes in all these three classes**  }    } |
| **50.** Create one interface with two abstract methods and implement those methods in one class and call them. |
| **51.**  interface SimCard{  void sms();  void call();  }  Class Test {  }  Class Demo{  main(){  **//implement SimCard in Test class and override methods**  **//call those methods from here**  }  } |
| **52.**  **Create Employee Object and display it’s data using toString that is only print e and it should display all the data of that object** |
| **53.**  **Create product class with id,name,price,qnty and create object with some data and display that data using toString** |
| **54.**  **Create Employee object and print its hashcode** |
| **55.**  **Create two employee objects with same data and compare two object(data)** |
| **56.**  **Create two product class objects and compare its data** |
| **57.**  **Create two employee objects and compare it’s references using == and equals** |
| **58.**  **Create an employee object with some data and create a clone of it and display data from the cloned object.** |
| **59.**  **Write a program which will explain finalize method** |
| **60.**  Class Test{  main(){  **//create five Employee objects here**  **//add those objects in ArrayList**  **//display object data from ArrayList using for loop**  }  } |
| **61.**  Class Test{  main(){  **//create five Employee objects here**  **//add those objects in ArrayList**  **//display object data from ArrayList using Iterator**  }  } |
| **62.** Class Test{  main(){  **//create five Employee objects here**  **//add those objects in Vector**  **//display object data from Vector using Enumeration**  }  } |
| **63.**  Class Test{  main(){  **//create five Employee objects here**  **//add those objects in ArrayList**  **//pass this ArrayList to display function**  }  Public static void display(){  **//do required changes to display method**  **//display ArrayList data here**  }  } |
| **64.**  public class Demo {  public static void main(String[] args) {  ArrayList<Integer> al = new ArrayList<Integer>();  al.add(10);  al.add(5);  al.add(7);  al.add(9);  **//sort list in ascending order**  }  } |
| **66.**  public class Demo {  public static void main(String[] args) {  ArrayList<Integer> al = new ArrayList<Integer>();  al.add(10);  al.add(5);  al.add(7);  al.add(9);  **//sort list in descending order (use reverse function)**  }  } |
| **67.**  public class Demo {  public static void main(String[] args) {  ArrayList<Integer> al = new ArrayList<Integer>();  al.add(10);  al.add(5);  al.add(7);  al.add(9);  **//sort list in descending order using Comparator**  }  } |
| **68.**  public class Demo {  public static void main(String[] args) {  ArrayList<Integer> al = new ArrayList<Integer>();  al.add(10);  al.add(5);  al.add(7);  al.add(9);  **//sort list in ascending order using Comparator**  }  } |
| **69.**  public class Demo {  public static void main(String[] args) {  ArrayList<Employee> al = new ArrayList<Employee>();  al.add(new Employee(102, "ram", 2000));  al.add(new Employee(103, "shyam", 3000));  al.add(new Employee(101, "sam", 10000));  al.add(new Employee(104, "sameer", 40000));  **//sort list in ascending order of id's using sort method and Comparable**  }  } |
| **70.**  public class Demo {  public static void main(String[] args) {  ArrayList<Employee> al = new ArrayList<Employee>();  al.add(new Employee(102, "ram", 2000));  al.add(new Employee(103, "shyam", 3000));  al.add(new Employee(101, "sam", 10000));  al.add(new Employee(104, "sameer", 40000));  **//sort list in descending order of id's using sort method and Comparable**  }  } |
| **71.**  public class Demo {  public static void main(String[] args) {  ArrayList<Employee> al = new ArrayList<Employee>();  al.add(new Employee(102, "ram", 2000));  al.add(new Employee(103, "shyam", 3000));  al.add(new Employee(101, "sam", 10000));  al.add(new Employee(104, "sameer", 40000));  **//sort list in ascending order of salary using sort method and Comparator**  }  } |
| **72.**  public class Demo {  public static void main(String[] args) {  ArrayList<Employee> al = new ArrayList<Employee>();  al.add(new Employee(102, "ram", 2000));  al.add(new Employee(103, "shyam", 3000));  al.add(new Employee(101, "sam", 10000));  al.add(new Employee(104, "sameer", 40000));  **//sort list in descending order of salary using sort method and Comparator**  }  }q |
| **73.**  public class Demo {  public static void main(String[] args) {  ArrayList<Integer> al = new ArrayList<Integer>();  al.add(10);  al.add(7);  al.add(5);  al.add(16);  al.add(7);  al.add(16);  **//remove duplicates from list and display (insertion order should not change)**  }  } |
| **74.**  public class Demo {  public static void main(String[] args) {  ArrayList<Integer> al = new ArrayList<Integer>();  al.add(10);  al.add(7);  al.add(5);  al.add(16);  al.add(7);  al.add(16);  ArrayList<Integer> al1 = new ArrayList<Integer>();  al1.add(100);  al1.add(7);  al1.add(5);  al1.add(56);  al1.add(17);  al1.add(26);  **//create a third arrayList al2 which should be the combination of both al and al1.**  }  } |
| **75.**  public class Demo {  public static void main(String[] args) {  ArrayList<Integer> al = new ArrayList<Integer>();  al.add(10);  al.add(7);  al.add(5);  al.add(16);  al.add(7);  al.add(16);  ArrayList<Integer> al1 = new ArrayList<Integer>();  al1.add(100);  al1.add(7);  al1.add(5);  al1.add(56);  al1.add(17);  al1.add(26);  **//create a third arrayList al2 which should be a combination of both al and al1. al2 should contain unique value.**    }  } |
| **76.**  public class Demo {  public static void main(String[] args) {  ArrayList<Employee> al = new ArrayList<Employee>();  al.add(new Employee(101, "sam", 1000));  al.add(new Employee(102, "sameer", 2000));  al.add(new Employee(103, "shyam", 3000));  al.add(new Employee(104, "ram", 4000));  al.add(new Employee(105, "ramesh", 5000));  al.add(new Employee(106, "sameera", 60000));  **//create another arrayList which contains only id of these Employees.**  }  } |
| **77.**  public class Demo {  public static void main(String[] args) {  ArrayList<Employee> al = new ArrayList<Employee>();  al.add(new Employee(101, "sam", 1000));  al.add(new Employee(102, "sameer", 2000));  al.add(new Employee(103, "shyam", 3000));  al.add(new Employee(104, "ram", 4000));  al.add(new Employee(105, "ramesh", 5000));  al.add(new Employee(106, "sameera", 60000));  **//create another three arrayLists which contain only id ,name ,salary of these Employees.**  **//example ArrayList<Integer> ids = …**  **//example ArrayList<String> names= ...**  }  } |
| **78.**  public class Demo {  public static void main(String[] args) {  ArrayList<Employee> al = new ArrayList<Employee>();  al.add(new Employee(101, "sam", 10000));  al.add(new Employee(102, "sameer", 2000));  al.add(new Employee(103, "shyam", 30000));  al.add(new Employee(104, "ram", 4000));  al.add(new Employee(105, "ramesh", 15000));  al.add(new Employee(106, "sameera", 60000));  **//create another arrayList which contains Employees whose salary is more than 5000**  **//display new list**  }  } |
| **79. Create HashSet of Integers and display data** |
| **80. Create HashSet of Employees and make sure duplicates are not there** |
| **81.**  class Product{  String name;  Int price;  Int quantity;  **//do required changes in Product class**  **}**  **Now create HashSet of Product and make sure duplicates are not there** |
| **82. Create hashMap with String key and Integer value and add some data into it. And display data from hashMap. Don’t use System.out.println(h)** |
| **83. Create HashMap<Employee,Integer> and data into it. Make sure duplicate keys are not added.** |
| **84. Create Map with sorting order of Keys** |
| **85. How to sort HashMap using it’s value** |

| **JDBC Questions**   1. Program which shows Driver registration and Connection creation 2. Programs for insert employee data into emp table using Statement 3. Take id and updated salary from user and update salary for that id using Statement 4. Take id from user delete record for that id using Statement 5. Take id from user and display record for that id using Statement 6. Display all the records from emp table using Statement 7. Programs for insert employee data into emp table using PreparedStatement 8. Take id and updated salary from user and update salary for that id using PreparedStatement 9. Take id from user delete record for that id using PreparedStatement 10. Take id from user and display record for that id using PreparedStatement 11. Display all the records from emp table using PreparedStatement 12. Take id and name from user and display record for that id and name using PreparedStatement 13. Display all the records whose salary is more than 1000 using PreparedStatement 14. Increase employee salaries by 500 whose salary is less than 10000 using PreparedStatement 15. Send insert,update,delete query using batch processing 16. Write a program which shows transaction management 17. Write a program which shows use of properties file |
| --- |