Assignments on Java Class

1) Create a class "Room" which will hold the "height", "width" and "breadth" of the room in three fields. This class also has a method "volume()" to calculate the volume of this room. Create another class "RoomDemo" which will use the earlier class, create instances of rooms, and display the volume of room.

2) Write a program to implement a class "student" with the following members.
Name of the student.
Marks of the student obtained in three subjects.
Function to assign initial values.
Function to compute total average.
Function to display the student's name and the total marks.
Write an appropriate main() function to demonstrate the functioning of the above.

3) Write a class "Box" that with three member-variables "height", "width" and "breadth". Write suitable constructors to initialize them. Add functions like "getVolume" and "getArea" that will return volume and surface area respectively. Instantiate two arbitrary boxes and then print their volume and surface area.

4) Implement a class for stack of integers using an array. Please note that the operations defined for a stack data structure are as follows: "push", "pop", "print".
There should be a constructor to create an array of integers; the size of the array is provided by the user.
Write a main() function to (i) create a stack to hold maximum of 30 integers; (ii) push the numbers 10, 20, 30, 15, 9 to the stack; (iii) print the stack; (iii) pop thrice and (iv) print the stack again.

5) Write a class "BankAccount" with the following instance variables:
AccountNumber (an integer), balance a floating-point number), and "ownerName" (a String). Write proper constructor for this class. Also write methods balance, add (to deposit an amount), and subtract (to withdraw an amount) and implement them. Now create another class "AccountManager" that contains an array of BankAccount. Write methods create (to create an account), delete(to terminate an account), deposit (to deposit an amount to an account) and withdraw (to withdraw an amount from an account). Also write a class "Bank", add main() function that creates an AccountManager and add 5 accounts. Now print the details of all accounts in this Bank.

6) Write a class to represent complex numbers with necessary constructors. Write member functions to add, multiply two complex numbers.
There should be three constructors: (i) to initialize real and imaginary parts to 0; (ii) to initialize imaginary part to 0 but to initialize the real part to user defined value; (iii) to initialize the real part and the imaginary part to user defined values.
Also, write a main() function to (i) create two complex numbers 3+2i and 4-2i; (ii) to print the sum and product of those numbers.

7) Implement a class for “Date”. Write member functions for (i) getting the previous day, (iv) getting the next day, (iii) printing a day
There should be four constructors: (i) day, month and year are initialized to 01, 01, 1970; (ii) day is initialized to user specified value but month and year are initialized to 01, 1970; (iii) day, month are initialized to user specified value but year is initialized to 1970; (iv) day, month and year are initialized to user defined values.
Also, write a main() function to (i) create a date object; (ii) print the next and the previous day.

8) Implement a class for a “Book”. Book contains a title (a String), a list of authors (array of authors), number of pages (an integer), price (floating point number), publisher (a String) etc. Write suitable constructor and accessor/modifier methods.
Implement a class for “Library”. A library contains a list of books (array of Book). Write add (to add a book) and remove (to delete a book) methods for library.
Write a main() function to create a “Library” and add five “Book” to library. Print the total price of all books.

9) Write a Java class “Employee” containing information name, id, address, salary etc. Write necessary constructor and read/write methods.
Create a class “Dept” that has a name, location etc. The “Dept” contains a number of “Employee”. Write methods “add” and “remove” to add and remove an employee to/from this department.
Write a main() function and create “Information Technology” department. Add five employees and print yearly expenditure for this department.

10) Implement a class for a “Student”. Information about a student includes name, roll no and an array of five subject names. The class should have suitable constructor and get/set methods.
Implement a class “TabulationSheet”. A tabulation sheet contains roll numbers and marks of each student for a particular subject. This class should have a method for adding the marks and roll no of a student.
Implement a class “MarkSheet”. A mark sheet contains marks of all subjects for a particular student. This class should have a method to add name of a student and marks in each subject.
Write a main() function to create three “Student” objects, Five “Tabulationsheet” objects for Five subjects and three “Marksheet” object for three students. Print the mark sheets.

Assignments on Java Inheritance

11) Create an abstract class “Publication” with data members ‘noOfPages’, ‘price’, ‘publisherName’ etc. with their accessor/modifier functions. Now create two sub-classes “Book” and “Journal”. Create a class Library that contains a list of Publications. Write a main() function and create three Books and two Journals, add them to library and print the details of all publications.

12) Write a class for “Account” containing data members ‘accountNumber’, ‘holderName’, ‘balance’ and add constructors and necessary accessor/modifier functions for these data members. Now create two class “SavingsAccount” and “CurrentAccount” extending from this

class. SavingsAccount will have a member variable ‘interestRate’ and member function ‘calculateYearlyInterest’. Write another class “Manager” that contains a list Account. Also write a main() function to create an instance of Manager class. Add two SavingsAccount and three CurrentAccount to Manager. Calculate interest of each SavingsAccount. Print the details of all accounts.

13) Implement a class for a “Person”. Person has data members ‘age’, ’weight’, ‘height’, ‘dateOfBirth’, ‘address’ with proper reader/write methods etc. Now create two subclasses “Employee” and “Student”. Employee will have additional data member ‘salary’, ‘dateOfJoining’, ‘experience’ etc. Student has data members ‘roll’, ‘listOfSubjects’, their marks and methods ‘calculateGrade’. Again create two sub-classes “Technician” and “Professor” from Employee. Professor has data members ‘courses’, ‘listOfAdvisee’ and their add/remove methods. Write a main() function to demonstrate the creation of objects of different classes and their method calls.

14) Create a base class “Automobile”. An Automobile contains data members ‘make’, ‘type’, ‘maxSpeed’, ‘price’, ‘mileage’, ‘registrationNumber’ etc. with their reader/writer methods. Now create two sub-classes “Track” and “Car”. Track has data members ‘capacity’, ‘hoodType’, ‘noOfWheels’ etc. Car has data members ‘noOfDoors’, ‘seatingCapacity’ and their reader/writer methods. Create a main() function to demonstrate this.

15) Implement the classes for the following inheritance hierarchies.
Create an interface “Shape” that contains methods ‘area’, ‘draw’, ‘rotate’, ‘move’ etc. Now create two classes “Circle” and “Rectangle” that implement this ‘Shape’ interface and implement the methods ‘area’, ‘move’, etc. Write a main() function to create two “Circle” and three “Rectangle”,then move them. Print the details of circles and rectangles before after moving them.

16) A bookshop maintains the inventory of books that are being sold at the shop. The list includes details such as author, title, publisher, cost and stock position. Whenever a customer wants a book, the sales person inputs the title and author and the system searches the list and displays whether it is available or not. If it is not, an appropriate message is displayed. If it is, then the system displays the book details and details and requests for the number of copies required. If the required copies are available, the total cost of the requested copies is displayed, otherwise the message “requires copies not in stock” is displayed. Design a system using a class called “Book” with suitable member methods and constructors.

17) Imagine a toll booth and a bridge. Cars passing by the booth are expected to pay an amount of Rs. 50/- as toll tax. Mostly they do but sometimes a car goes by without paying. The toll booth keeps track of the number of the cars that have passed without paying, total number of cars passed by, and the total amount of money collected. Execute this with a class called “Tollbooth” and print out the result as follows:
The total number of cars passed by without paying.
Total number of cars passed by.
Total cash collected.

18) Write two interfaces “Fruit” and “Vegetable” containing methods ‘hasAPeel’ and ‘hasARoot’. Now write a class “Tomato” implementing Fruit and Vegetable. Instantiate an object of Tomato. Print the details of this object.

Assignments on Java Thread

19) Write a program to create two threads. Print “In main thread” in main thread and “In child thread” in child thread.

20) Create two threads and call them EvenThread and OddThread. EvenThread will print number as 2 4 6 8 10... and OddThread will print number as 1 3 5…. Now synchronize these two thread to get the output as 1 2 3 4 5 6 7 8.

21) Consider the following series
x = 1+1/1! +1/2! + 1/3! + ……1/10!
Create two threads t1 & t2. t1 will generate the denominators and t2 will form the term and add them up. Finally print the result.

22) Consider a file that contains a number of integers. Create two threads. Call them ‘producer’ and ‘consumer’ thread. Producer thread will be reading the integers from the file continuously while consumer thread will add them up. Use proper synchronization mechanism if needed.

23) Consider the series 1+2+3+…+100. This can be considered as (1+3+5+…+99)+(2+4+6+…+100). Create two threads to compute two series in parallel (do not use simplified equation). Finally print the final sum.

24) Consider the following parallel binary search algorithm for series a1, a2…an sorted in increasing order such that n mod 10 = 0. Element to be searched is e.

a) Create n/10 threads t1, t2,..,tn/10.
b) Distribute the numbers among threads such that ti will have numbers ai, ai+1, ….a2i-1.
c) Distribute the element e to all threads.
d) Each thread searches the element e in its sub-array using binary search algorithm.

25) Write a Java program using threading technology and print the thread index and location where the element has been found.

Java Swing

26) Display the message “Welcome to Java swing programming” in a message dialog box.

27) Write a program to calculate the squares and cubes of the numbers from 1 to 10 and print the result in a tabular format on a dialog box.

28) Write a program using swing technology that reads first and last name from the user as two separate inputs and concatenates the first and last name separated by a space. Display in a message dialog the concatenated name.

29) Write a program to add two numbers and print the result. The program will work as follows: A window will contain three text boxes with captioned “number 1”, “number 2”, “result” respectively and a command button with captioned “add”. First two text boxes will be used to input two numbers and the third text box to display the result. Result will be displayed when command button will be pressed.

30) Write a program to design a simple calculator using Java swing technology. Your calculator should be able to perform simple operations such as addition, multiplication, subtraction and division.

Socket

31) Create two processes P1 and P2. Call the process P1 as client process and the process P2 as server process. Now client process reads an integer form the keyboard, sends it to the server process using a socket. Server process now calculates the factorial of the number sent by the client process and sends it back through the socket. Client process in turn accepts the factorial of the number it sent and prints it.

RMI

32) Write a Java interface that accepts a set of numbers and sorts it. Write a server program to implement it. Now create a client process that sends a set of numbers to the server process through Java RMI technology. Server process in turn sorts the numbers. Client process then prints the set of numbers in sorted order.