# CORE JAVA LAB TEST

NAME:-SAURABH AMBADAS PALE

Design application for scholarship program in college.

Saurabh A. Pale

① College.java

```
package core-java-lab-test;
public interface College {
// Q.1 Create public interface college having filed like college-
//   name, address, phone.
    public static final String clg-name = "CDAC";
    public static final String clg-address = "NOIDA";
    public static final int clg-number = 2423;

Q.3 public static void printMessage()
    {
        System.out.println("WELCOME!");
    }
Q.2 abstract void getDetails()
    abstract void showDetails();

}
```

② Student.java

```
package core-java-lab-test;
import java.util.Scanner;
public class Student implements College {
 int Count = 0;
Q.4 private static String studentname;
    private static int rollno;
    private static float marks;
    private static float attendance;

Q.5 Student()
    {
        System.out.println("College Management System");
    }
```

Q-5 SAVING A FILE

```
Student ( String Studentname , int rollno , int marks , float attendance)
  {
     this. Studentname = Studentname;
     this. rollno = rollno ;
     this. marks = marks ;
     this. attendance = attendance ;
  }
```

Q6

```
public void getDetails() {
      College.printMessage();
}
public String getStudentname() {
      return studentname;
  }
public int getRoll() {
      return rollno;
}
public int getMarks() {
     return marks ;
}
public float getAttendance() {
      return attendance ;
  }
  public void showDetails() {
      System.out.println("College name " + (clg_name);
      System.out.println("College Address " + (clg_address);
      System.out.println(" College number " + clg_number);
  }
}
```

⑧ Scholarship.java

```
package Core_java_Lab_test;
import java.util.Scanner;
public class Scholarship {
    public static void main (String[] args) {
            Count = 0;
        Student S1 = new Student();
         S1.getDetails();
         S1.showDetails();

      Scanner S = new Scanner (System.in);
      Student Ss[] = new Student[5];
```

Q.7.

```
for (int i=0 ; i<5 ; i++)
    {
     System.out.println("Enter Student Name");
      String Studentname = S.next();
     System.out.println("Enter Student Rollno : ");
      int rollno = S.nextInt();
      System.out.println("Enter Student Marks :");
      int marks = S.nextInt();
      System.out.println("Enter Student Attendance :");
      float attendance = S.nextFloat();
      System.out.println("* * * * * * * * * * * * * * * * * * * * ");
     Ss[i] = new Student (Studentname, rollno, marks, attendance);
```

Q.8.

```
    if (marks > 90 && attendance > 80)
     {
       System.out.println("Student is Eligible for Scholarship");
       System.out.println("* * * * * * * * * * * * * * * * * * * *");
        Count ++;

     }
```

```
        else
        {
            System.out.println("Not Eligible for Scholarship");
        }
    }
    for(int i=0; i<5; i++)
    {
        System.out.println("Student Name "+" "+sa[i].getStu-
                dentname() + "\n" + "Student Rollno"
            +" "+sa[i].getRollno() + "\n" + "\n"
            "Marks"+" "+sa[i].getMarks()+"\n"
        + "Attendance" +" "+sa[i].getAttendance());
    }
    System.out.println("Count of Student Eligible for Scholarship"
                    +count);
    }
}
```

e.g.

(Q 10) What is OOP?

⇒ ① Object-oriented programming is a computer programming model that organize software design around data or objects, rather than function & logic. An object can be defined as a data field that has unique attributes & behavior.

② It is a programming methodology to oraganize complex program in to simple program in terms of class & object such methodology is called Oops.

③ Oops is a programming approach which revolves around the concept of "Object".

④ Languages which support abstraction, encapsulation, polymorphism & inheritance are called Oops language.

⑤ Pillars of OOps [major pillars]

- ⓐ Abstraction
- ⓑ Encapsulation
- ⓒ ~~Encapsulation~~ Polymorphism.
- ⓓ Hierarchy / Inheritance

* ⓐ Abstraction:- It is used to provide essential features of an object to a user & hiding its background details.

① Outer behaviour
② Hiding the details
③ function call is also known as Abstraction.

eg. The man pressing the accelerators will increases speed of a car or applying brakes will stop the car but he does not know about how on pressing accelerator the speed is actually increasing, he doesnot know about the inner mechanism of the car.

* (b) Encapsulation :-

① It binds the variables & methods together into a single unit called class.

② Inner behaviour

③ Function definition is also known as as Encapsulation

④ Implementation of abstraction is Encapsulation.

⑤ Binding of [Code + Data] is scope

Code = member function

data = data member.

e.g.

① School bag is one of the most real example of Encapsulation. School bag can keep our books, pens etc.

* (c) Polymorphism :- One thing many forms

① One interface having multiple forms is called as polymorphism.

* two types of polymorphism.

① Compile time polymorphism. — (a) function Overloading
(b) Operator Overloading

② Run time polymorphism. — (a) function Overriding.

e.g.

A person at the same time can have different characteristic. Like a man at the same time is a father, a husband, an employee. So that same person possesses different behavior in different situations.

(d) Inheritance :- Reusability
Hierarchy is ranking or ordering of abstraction.

(a)
(1) Inheritance is the capability of one class to inherit capabilities or properties from another class in Java.

e.g. (1) The class "Car" inherits its properties from the class "Automobiles" which inherits some of its properties from another class "Vehicles".

* Types of Java Inheritance
(a) Single Inheritance
(b) Multilevel Inheritance
(c) Hierarchical Inheritance
(d) Multiple Inheritance
(e) Hybrid Inheritance.

* Saurabh A. Pale *

Q. 8 11) What is Serialization?

→ Serialization is the conversion of the state of an object into a byte stream. It is the conversion of a java object into a static stream of bytes.
Which we can then save to a database or transfer over a network.
When you are transferring information from one system another in a network, the information is transmitted in bytes.

e.g. you are going to visit your home after 20 yrs and you want the same things, same arrangement & many more things which you left 20 yrs ago in same condition as where u left..

Q 12) What is Synchronization? ⓐ * Sreenath A. Pai *

⇒ · It is capability to control the access of multiple threads to any shared resource. In the multithreading concept, multiple threads try to access the shared resources at a time to produce inconsistent result. The synchronization is necessary for reliable communication between threads.

* Why are we Synchronization.
  ① It helps in preventing thread interference.
  ② It helps to prevent concurrency problems.

* Types of Synchronization
  ① Process Synchronization.
  ② Thread Synchronization.

e.g. ① Suppose a thread in a program is reading a record from a file while another thread is still writing the same file. In this situation, the program may produce undesirable output.

Slide

Q 13) Difference between hashset, treeset & hash map?

⇒

| | Hashset | Treeset | Hash map |
|---|---|---|---|
| 1. | • Hash set is implemented using HashTable | • The tree set is implemented using a tree structure. | • The Hash map is implemented of Map interface. |
| 2. | • HashSet allows a null object | • The tree set doesn't allow the null object. It throughs the null pointer exception | • Single null key & and any no. of null value can be inserted in hashmap without any restriction. |
| 3. | • HashSet does not maintain any order | • Tree set maintain an object in sorted order | • Hashmap due to its unique key is faster in retrieval of element during its iteration. |

Q(14) What is Compile time Polymorphism & Runtime Polymorphism ?

⇒ Polymorphism :- The compiler check the type of reference in the object & ont the type of objects.

① One many forms

② Same function is used to perform different kind of operations.

* 2 types of polymorphism

↳ ① Compile time polymorphism

ⓐ function overloading

ⓑ operator overloading

↳ ② Run time polymorphism

ⓐ function overriding.

ⓐ Compile time Polymorphism :-

| | Compile time Polymorphism | Run time Polymorphism |
|---|---|---|
| ① | Compile time Polymorphism means binding is occuring at Compile time | ① Run time Polymorphism where at run time we came to know which method is going to invoke. |
| ② | It can be achieved through static binding | ② It can be achieved through dynamic binding |
| ③ | Inheritance is not involved | ③ Inheritance is involved |
| ④ | Method overloading is an example of Compile time polymorphism | ④ Method overriding is an example of runtime polymorphism. |

Q.15> What is Collection in Java ? Difference between set, list & map & queue. (5)

[Saurabh A. Pole]

⇒ Collection in Java :→

① The Collection in Java is a framework that provides an architecture to store & manipulate the group of objects.

② Java Collection can achieve all the operation that you perform on a data such as searching, sorting, insertion, manipulation & deletion.

③ A Collection represents a single unit of objects. i.e group.

* ① Set :→ ① Set doesn't allow duplicate elements.

② Set do not maintain any insertion order.

③ But in Set almost only one null value.

④ Set implementation classes are HashSet, LinkedHashSet & TreeSet.

* ② List :→ ① The list interface allows duplicate elements

② The list maintains insertion order.

③ We can add any no. of null values.

④ List implementation classes are Array list, LinkedList.

* ③ map :→ ① The map does not allow duplicate elements

② The map also does not maintain any insertion order.

③ The map allows a single null key at most & any no. of null values.

* @queue :- ① Queue does allow duplicate elements.

③ All elements get inserted at the "end" & removed from the "begining" of the queue.

② Queue interface in java collection has two implementation - LinkedList & PriorityQueue, these two classes implements Queue interface.

Author

* Saurabh Ambadas Pole *