RASHTREEYA SIKSHANA SAMITHI TRUST

RV COLLEGE OF ENGINEERING®

(Autonomous Institution Affiliated To VTU, BELAGAVI)

RV Vidyaniketan Post, Mysuru Road,

BENGALURU - 560 059

Department of Artificial Intelligence and Machine Learning

Lab Manual for IV Semester

Object Oriented Programming Using Java (21CS49)

LABORATORY SOLTION MANUAL

Prof. Narasimha Swamy S

June 2023

PROGRAM 1

Program 1a: Create a Java class called Complex with the following details as member variables within it. (i) Real (ii) Imaginary Develop a Java program to perform addition and subtraction of two complex numbers by using the method add () and subtract () respectively, by passing object as parameter and display result using method display (). Initialize the real and imaginary values of the complex number using parameterized constructor. Also demonstrate overloading constructors and methods.

```java
package Lab1;

class Complex
{
	int real,imag;
	
	Complex()
	{
		real=0;
		imag=0;
	}
	
	Complex(int r,int i)
	{
		this.real=r;
		this.imag=i;
	}
	
	public Complex add(Complex n1,Complex n2)
	{
		Complex temp=new Complex(0,0);
		temp.real=n1.real+n2.real;
		temp.imag=n1.imag+n2.imag;
		return temp;
	}
	
	public Complex sub(Complex n1,Complex n2)
	{
		Complex temp=new Complex(0,0);
		temp.real=n1.real-n2.real;
		temp.imag=n1.imag-n2.imag;
		return temp;
	}
	
	public void display(Complex n)
	{
		System.out.println(n.real+"+i"+n.imag);
	}
}

public class Main
{
	public static void main(String[] args)
	{
		Complex ob=new Complex();
		Complex ob1=new Complex(10,20);
		Complex ob2=new Complex(30,40);
		Complex a=ob.add(ob1,ob2);
		Complex s=ob.sub(ob2,ob1);
		System.out.print("After addition: ");
		ob.display(a);
		System.out.print("After subtraction: ");
		ob.display(s);
	}
}
```

Program 1b: Design an Address class with member variables Street num, city, state and country and appropriate constructor. Design a student class with constructor (Student (String USN, String Name, Address addr)), College class with constructor (College (String Name, Address addr)) and Employee class with constructor (Employee (String EmpID, String Name, Address addr)). Write a Java program to create 'n' Student objects, College Objects and Employee objects and print the student, college and employee addresses respectively and demonstrate passing of object as a parameter to the constructor

```
package Lab1b;

import java.util.*;

class Address
{
	int streetnum;
	String city;
	String state;
	String country;

	Address(int sn,String c,String s,String co)
	{
		streetnum=sn;
		city=c;
		state=s;
		country=co;
	}

	int getStreetnum()
	{
		return streetnum;
	}

	String getCity()
	{
		return city;
	}

	String getState()
	{
		return state;
	}

	String getCountry()
	{
		return country;
	}

	void printAddress()
	{
		System.out.println("Street number: 
"+this.getStreetnum());
		System.out.println("City: "+this.getCity());
```

```java
		System.out.println("State: "+this.getState());
		System.out.println("Country: "+this.getCountry());
	}
}

class Student
{
	String USN;
	String name;
	Address addr;

	Student(String u,String n,Address a)
	{
		USN=u;
		name=n;
		addr=a;
	}
}

class College
{
	String name;
	Address addr;

	College(String n,Address a)
	{
		name=n;
		addr=a;
	}
}

class Employee
{
	String EmpID;
	String name;
	Address addr;

	Employee(String e,String n,Address a)
	{
		EmpID=e;
		name=n;
		addr=a;
	}
}

public class Main
{
	public static void main(String[] args)
	{
		Scanner sc=new Scanner(System.in);
		System.out.println("1.Student");
		System.out.println("2.College");
		System.out.println("3.Employee");
```

```
          System.out.println("Enter your choice");
          int ch=sc.nextInt();
          int n=0;
          if(ch==1)
          {
               System.out.println("Enter the number of students");
               n=sc.nextInt();
               String n1=sc.nextLine();
               Student st[]=new Student[n];
               for(int i=1;i<=n;i++)
               {
                    System.out.println("Enter the name of the
                    student");
                    String na=sc.nextLine();
                    System.out.println("Enter the USN of the
                    student");
                    String u=sc.nextLine();
                    System.out.println("Address");
                    System.out.println("Enter street number");
                    int sn=sc.nextInt();
                    System.out.println("Enter city");
                    String c1=sc.nextLine();
                    String c=sc.nextLine();
                    System.out.println("Enter state");
                    String s=sc.nextLine();
                    System.out.println("Enter country");
                    String co=sc.nextLine();
                    Address a=new Address(sn,c,s,co);
                    st[i]=new Student(u,na,a);
               }
               System.out.println();
               for(int i=1;i<=n;i++)
               {
                    System.out.println("Student "+i+" details");
                    System.out.println("Name: "+st[i].name);
                    System.out.println("USN: "+st[i].USN);
                    st[i].addr.printAddress();
                    System.out.println();
               }
          }
          else if(ch==2)
          {
               System.out.println("Enter the number of colleges");
               n=sc.nextInt();
               String n1=sc.nextLine();
               College st[]= new College[n];
               for(int i=0;i<n;i++)
               {
                    System.out.println("Enter the name of the
college");

                    String na=sc.nextLine();
                    System.out.println("Address");
                    System.out.println("Enter street number");
```

```
				int sn=sc.nextInt();
				System.out.println("Enter city");
				String c1=sc.nextLine();
				String c=sc.nextLine();
				System.out.println("Enter state");
				String s=sc.nextLine();
				System.out.println("Enter country");
				String co=sc.nextLine();
				Address a=new Address(sn,c,s,co);
				st[i]=new College(na,a);
			}
			System.out.println();
			for(int i=0;i<n;i++)
			{
				System.out.println("College "+(i+1)+"
details");

				System.out.println("Name: "+st[i].name);
				st[i].addr.printAddress();
				System.out.println();
			}
		}
		else if(ch==3)
		{
			System.out.println("Enter the number of employees");
			n=sc.nextInt();
			String n1=sc.nextLine();
			Employee st[]=new Employee[n];
			for(int i=1;i<=n;i++)
			{
				System.out.println("Enter the name of the
Employee");

				String na=sc.nextLine();
				System.out.println("Enter the employee ID of
the Employee");

				String u=sc.nextLine();
				System.out.println("Address");
				System.out.println("Enter street number");
				int sn=sc.nextInt();
				System.out.println("Enter city");
				String c1=sc.nextLine();
				String c=sc.nextLine();
				System.out.println("Enter state");
				String s=sc.nextLine();
				System.out.println("Enter country");
				String co=sc.nextLine();
				Address a=new Address(sn,c,s,co);
				st[i]=new Employee(u,na,a);
			}
			System.out.println();
			for(int i=1;i<=n;i++)
			{
				System.out.println("Employee "+i+" details");
				System.out.println("Name: "+st[i].name);
```

```
                    System.out.println("Employee ID: 
"+st[i].EmpID);
                    st[i].addr.printAddress();
                    System.out.println();
               }
          }
          else
          {
               System.out.println("Invalid Input");
          }
          sc.close();
     }
}
```

Program 2a: Design a base class Circle with member variables (radius and color) of type double, methods (getRadius(), getArea()) and constructors (Circle(radius), Circle(radius, color)). Derive subclass called Cylinder from the superclass Circle with member variable (height) of type double, public methods (getHeight(), getVolume(), getArea()) and its constructors(Cylinder(height, radius), Cylinder(height, radius,color)). Create the two instances of cylinder and print similar cylinders if the area, volume and color of cylinders are same. Demonstrate the code reuse and polymorphism properties of Object-oriented programming by inheriting the constructors and methods of the base class.

```
package Lab2a;

import java.util.*;
class Circle
{
	double radius;
	String color;

	Circle()
	{
		radius=0;
	}
	Circle(double r)
	{
		radius=r;
	}
	Circle(double r,String c)
	{
		radius=r;
		color=c;
	}
	double getRadius()
	{
		return radius;
	}
	double getArea()
	{
		return Math.PI*radius*radius;
	}
	String getColor()
	{
		return color;
	}
}

class Cylinder extends Circle
{
	double height;
```

```java
	Cylinder(double height,double radius)
	{
		super(radius);
		this.height=height;
	}
	Cylinder(double height,double radius,String color)
	{
		super(radius,color);
		this.height=height;
	}

	double getHeight()
	{
		return height;
	}
	double getArea()
	{
		return 2*Math.PI*super.radius*(super.radius+height);
	}
	double getVolume()
	{
		return super.getArea()*height;
	}
}

public class Main
{

	public static void main(String[] args)
	{
		Scanner sc=new Scanner(System.in);
		System.out.println("Cylinder 1");
		System.out.println("Enter the radius of the cylinder");
		double r=sc.nextDouble();
		System.out.println("Enter the height of the cylinder");
		double h=sc.nextDouble();
		System.out.println("Enter the color of the cylinder");
		String c=sc.nextLine();
		c=sc.nextLine();
		Cylinder c1=new Cylinder(h,r,c);

		System.out.println("Cylinder 2");
		System.out.println("Enter the radius of the cylinder");
		r=sc.nextDouble();
		System.out.println("Enter the height of the cylinder");
		h=sc.nextDouble();
		System.out.println("Enter the color of the cylinder");
		c=sc.nextLine();
		c=sc.nextLine();
		Cylinder c2=new Cylinder(h,r,c);
		if(c1.getArea()==c2.getArea() && 
c1.getVolume()==c2.getVolume() && 
c1.getColor().equalsIgnoreCase(c2.getColor()))
		{
			System.out.println("The Cylinders are similar");
		}
		else
		{
			System.out.println("The Cylinders are not similar");
		}
		sc.close();
	}
}
```

PROGRAM 3

Program 3a: Create a class FourthSemester. Put this class into a package called AIML. Define a method WelcomeMsg() which prints a line “Welcome to AIML Department 4th Semester Young Budding Engineers”. Create a class AIML-Dept. Put this class into a package called RVCE. Inherit the class FourthSemester in AIML package to AIMLDept class in RVCE package and call WelcomeMsg() method to display welcome message and also verify public method Overriding, Private method overriding and default method overriding from different packages in java with the same program

```java
package AIML;

public class FourthSemester
{
	protected void WelcomeMsg()
	{
		System.out.println("Welcome to AIML Dept-4th Semester 
Young Budding Engineers");
	}
}


package RVCE;

public class AIMLDept extends AIML.FourthSemester
{
	public static void main(String[] args)
	{
		AIMLDept ob=new AIMLDept();
		ob.WelcomeMsg();
	}
}
```

Program 3b: Create two classes called Lion and Snake that implements all the methods defined in an interface Animal. Declare **eat () and sound ()** methods in Animal interface and display eating habits and sound made by that particular animal respectively. Create an interface called **Tired Animal**. In Tired Animal interface add method definition to an existing interface by extending Animal interface to verify Extending Interface concept in java.

Note: Lion and Snake implement the required eat () method and has some of its own methods and instance variables

```java
package Lab3b;

interface Animal
{
	void eat();
	void sound();
}

interface TiredAnimal extends Animal
{
	void sleep();
}

class Lion implements Animal
{
	@Override
	public void eat()
	{
		System.out.println("Lions are carnivores and eat meat");
	}

	@Override
	public void sound()
	{
		System.out.println("Lions roar loudly");
	}

	public void body()
	{
		System.out.println("Lions have 4 legs");
	}
}

class Snake implements TiredAnimal
{
	@Override
	public void eat()
	{
		System.out.println("Snakes swallow their prey whole");
	}

	@Override
	public void sound()
	{
		System.out.println("Snakes hiss to communicate");
	}
	
	@Override
	public void sleep()
	{
		System.out.println("Snakes sleep after eating their 
prey");
	}
	
	public void body()
	{
		System.out.println("Snakes do not have legs");
	}
}

public class Main
{
	public static void main(String[] args)
	{
		System.out.println("-------------------------------------
--------------------------");
		Lion lion=new Lion();
		lion.eat();
		lion.sound();
		lion.body();
		System.out.println("-------------------------------------
--------------------------");
		Snake snake=new Snake();
		snake.eat();
		snake.sound();
		snake.sleep();
		snake.body();
		System.out.println("-------------------------------------
--------------------------");
	}
}
```

PROGRAM 4

Program 4a: Design and implement a Java program for the following requirements:

1. An Exception class called Demonetization Exception which returns the statement that says "Deposit of Old currency of (Rs________) crosses Rs. 5,000 and cannot be Deposited"
2. A class called 'Account' that creates account with 500 Rs minimum balance with following methods.
   a. Deposit (Amount, CurrencyType) method to deposit amount. This class should handle "Demonetization Exception" and print the message defined in this Exception class. If a currency type is "OLD" and the amount is greater than 5,000 then throw the Demonetization Exception, otherwise update the balance.
   b. CurrBalance() method that displays balance amount in the account.
   c. Withdraw(amount) method to withdraw amount and update the balance. Use proper control structure to check Balance should not go less than 500.
3. 'Customer' class that creates Account object and call the methods deposit (), withdraw () and currBalance() based on the user choice.

```
package Lab4a;

class DemonetizationException extends Exception
{
	DemonetizationException(int amount)
	{
		super("deposit of old currency of Rs."+ amount + " 
crosses Rs.5000");
	}
}

class account
{
	int balance;
	account()
	{
		balance=500;
	}

	void deposit(int amount,String currencyType)
	{
		try
		{
			if(currencyType.equalsIgnoreCase("old") && 
amount>5000)
			{
				throw new DemonetizationException(amount);
			}
			else
			{
				balance+=amount;
				System.out.println("Deposit Successful!");
```

```
                }
            }
            catch (DemonetizationException e)
            {
                System.out.println(e.getMessage());
            }
        }
        void CurrBalance()
        {
            System.out.println("The current balance is Rs."+balance);
        }
        void withdraw(int amount)
        {
            if(balance-amount<500)
            {
                System.out.println("Withdrawal failed! Balance 
should not go below 500");
            }
            else
            {
                balance-=amount;
                System.out.println("Withdrawal Successful");
            }
        }
}


public class Main
{

        public static void main(String[] args)
        {
            account ac=new account();
            ac.deposit(400,"old");
            ac.withdraw(300);
            ac.CurrBalance();
            ac.deposit(6000,"old");
            ac.withdraw(7000);
        }
}
```

Program 5a: Design and develop a Java program for the fruit market problem. The farmer will be able to produce different types of fruits (apple, orange, grape, and watermelon), and put them in the market to sell. The market has limited capacity and farmers have to stand in a queue if the capacity is exceeded to sell their fruits. Consumers can come to the market any time and purchase their desired fruits; and if the fruits they want to buy runs out, they are willing to wait until the supply of that kind is ready. Examine and formulate an approach to address this problem and implement the same using Java constructs for programming.

```java
package Lab5;

import java.util.*;
class Market
{
	private int fruitsNumber;
	private ArrayList<String> fruits;

	public Market(int fruitsNumber)
	{
		this.fruitsNumber=fruitsNumber;
		this.fruits=new ArrayList<>(fruitsNumber);
	}

	public synchronized boolean isFull()
	{
		return fruits.size()==fruitsNumber;
	}

	public synchronized boolean isEmpty()
	{
		return fruits.isEmpty();
	}

	public synchronized void farmer(String fruit)
	{
		if(isFull())
		{
			try
			{
				wait();
			}
			catch(InterruptedException e)
			{
				e.printStackTrace();
			}
		}
		fruits.add(fruit);
		notify();
	}

	public synchronized String consumer()
	{
		if(isEmpty())
		{
			try
			{
				wait();
			}
			catch(InterruptedException e)
			{
				e.printStackTrace();
			}
		}
		String consumedFruit=fruits.remove(0);
		notify();
		return consumedFruit;
	}
}

public class Main
{
	public static void main(String[] args)
	{
		Market m=new Market(5);
		Thread farmerThread=new Thread(new Runnable()
		{
			@Override
			public void run()
			{
				for(int i=1;i<=10;i++)
				{
					String fruit="Fruit "+i;
					m.farmer(fruit);
					System.out.println("Farmer added: 
"+fruit);
				}
			}
		});

		Thread consumerThread=new Thread(new Runnable()
		{
			@Override
			public void run()
			{
				for(int i=1;i<=10;i++)
				{
					String consumedFruit=m.consumer();
					System.out.println("Consumer consumed: 
"+consumedFruit);
				}
			}
		});

		farmerThread.start();
```

```
            consumerThread.start();
        }
    }
```

PROGRAM 6

Program 6a: Write a Java program to create a new array list, add some colors (string) and perform the following operations:

1. Add elements of List to ArrayList
2. Copy ArrayList to Array
3. Reverse ArrayList content
4. Get Sub list from an ArrayList.
5. To sort a given ArrayList
6. Clone an ArrayList to another ArrayList

```
package Lab6;

import java.util.*;

public class Main
{
	public ArrayList<String> list=new ArrayList<String>();

	public static void main(String[] args)
	{
		Main op=new Main();
		Scanner sc=new Scanner(System.in);
		for(;;)
		{
			System.out.println("------ Choice List ------");
			System.out.println("1.Insert");
			System.out.println("2.Copy ArrayList to Array");
			System.out.println("3.Reverse ArrayList contents");
			System.out.println("4.Get sub list from ArrayList");
			System.out.println("5.Sort the ArrayList");
			System.out.println("6.Clone the ArrayList to another 
ArrayList");
			System.out.println("Enter your choice");
			int ch=sc.nextInt();
			String co=sc.nextLine();
			switch(ch)
			{
				case 1:
					op.list.add("BLACK");
					op.list.add("WHITE");
					op.list.add("BLUE");
					op.list.add("GREEN");
					op.list.add("RED");
					op.list.add("ORANGE");
					op.list.add("YELLOW");
					op.list.add("PURPLE");
					op.list.add("GREY");
					for(String color : op.list)
					{
						System.out.println(color);
					}
					break;

				case 2:
					String [] st=new String[op.list.size()];
					op.list.toArray(st);
					System.out.println("Copied Array");
					for(String color : op.list)
					{
						System.out.println(color);
					}
					break;

				case 3:
					Collections.reverse(op.list);
					System.out.println("Reversed 
					ArrayList:");
					for(String color : op.list)
					{
						System.out.println(color);
					}
					break;

				case 4:
					ArrayList<String> list1=new 
ArrayList<>(op.list.subList(2,4));
					System.out.println("Sublist:");
					for(String color : list1)
					{
						System.out.println(color);
					}
					break;

				case 5:
					Collections.sort(op.list);
					System.out.println("Sorted ArrayList:");
					for(String color : op.list)
					{
						System.out.println(color);
					}
					break;

				case 6:
					ArrayList<String> 
					list2=(ArrayList<String>)op.list.clone();
					System.out.println("Cloned ArrayList:");
					for(String color : list2)
					{
						System.out.println(color);
					}
					break;

				default:
					sc.close();
```

```
                    return;
                }
            }
        }
    }
```