1.Bus Ticket Management System using Inheritance and Interface

```
import java.util.*;

interface Seatmgt {
	void showavail();
	void updateseats();
}

class Bus {
	protected String busno;
	protected String route;
	protected String bustype;
	protected int totalseats;

	public Bus(String busno, String route, String bustype, int 
totalseats) {
		this.busno = busno;
		this.route = route;
		this.bustype = bustype;
		this.totalseats = totalseats;
	}

	void showdetails() {
		System.out.println("\nBus number : "+busno+"\nRoute : 
"+route+"\nBus Type : " +bustype+"\nAvailable Seats : "+totalseats);
	}
}

class Booking extends Bus {
	protected String passengername;
	protected int bookedseats;

	public Booking(String busno,String route, String bustype, int 
totalseats) {
		super(busno,route,bustype,totalseats);
	}

	void bookticket(String passengername, int seats) {
		if(seats <= totalseats) {
			this.passengername = passengername;
			this.bookedseats = seats;
			totalseats -= seats;
			System.out.println("Ticket booked");
		}
		else {
```

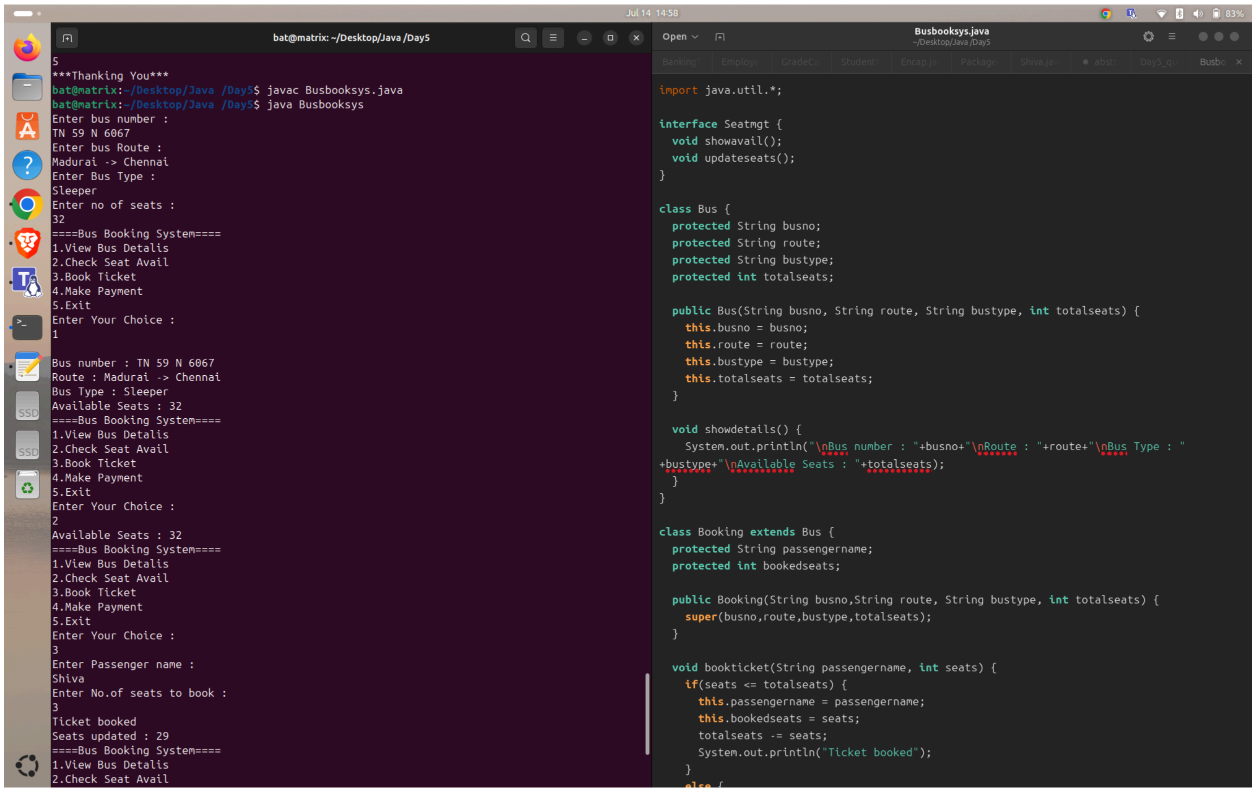
```
                System.out.println("Seats unavailable");
            }
        }
    }

    class Payment extends Booking implements Seatmgt {
        public Payment(String busno,String route, String bustype, int
    totalseats) {
            super(busno,route,bustype,totalseats);
        }

        public void makepayment(double amount) {
            System.out.println("Paid successfully "+amount+" for
    "+passengername);
        }

        public void showavail() {
            System.out.println("Available Seats : "+totalseats);
        }

        public void updateseats() {
            System.out.println("Seats updated : "+totalseats);

        }
    }

    public class Busbooksys {
        public static void main(String[] args) {
            Scanner sc = new Scanner(System.in);

            System.out.println("Enter bus number : ");
            String busno = sc.nextLine();
            System.out.println("Enter bus Route : ");
            String route = sc.nextLine();
            System.out.println("Enter Bus Type : ");
            String bustype = sc.nextLine();
            System.out.println("Enter no of seats : ");
            int seats = sc.nextInt();

            Payment bk = new Payment(busno,route,bustype,seats);

            int number;

            do{
                System.out.println("====Bus Booking
System====\n1.View Bus Detalis\n2.Check Seat Avail\n3.Book
Ticket\n4.Make Payment\n5.Exit\nEnter Your Choice : ");
                number = sc.nextInt();


                switch(number){
                  case 1:
                    bk.showdetails();
                    break;
                  case 2:
                    bk.showavail();
                    break;
                  case 3:
                    sc.nextLine();
                    System.out.println("Enter Passenger name : ");
                    String name = sc.nextLine();
                    System.out.println("Enter No.of seats to book
:");

                    int bookseats = sc.nextInt();
                    bk.bookticket(name,bookseats);
                    bk.updateseats();
                    break;
                  case 4:
                    System.out.println("Enter Payment Amount : ");
                    double amount = sc.nextDouble();
                    bk.makepayment(amount);
                    break;
                  case 5:
                    System.out.println("***Thanking You***");
                    break;
                  default :
                     System.out.println("Enter only valid
options...");
                }
           }while(number != 5);
           sc.close();
      }
}
```

Output:

```
Enter bus number :
TN 59 N 6067
Enter bus Route :
Madurai -> Chennai
```

```
Enter Bus Type :
Sleeper
Enter no of seats :
32
====Bus Booking System====
1.View Bus Detalis
2.Check Seat Avail
3.Book Ticket
4.Make Payment
5.Exit
Enter Your Choice :
1

Bus number : TN 59 N 6067
Route : Madurai -> Chennai
Bus Type : Sleeper
Available Seats : 32
====Bus Booking System====
1.View Bus Detalis
2.Check Seat Avail
3.Book Ticket
4.Make Payment
5.Exit
Enter Your Choice :
2
Available Seats : 32
====Bus Booking System====
1.View Bus Detalis
2.Check Seat Avail
3.Book Ticket
4.Make Payment
5.Exit
Enter Your Choice :
3
Enter Passenger name :
Shiva
Enter No.of seats to book :
3
Ticket booked
Seats updated : 29
====Bus Booking System====
1.View Bus Detalis
2.Check Seat Avail
3.Book Ticket
4.Make Payment
```

```
5.Exit
Enter Your Choice :
4
Enter Payment Amount :
1500
Paid successfully 1500.0 for Shiva
====Bus Booking System====
1.View Bus Detalis
2.Check Seat Avail
3.Book Ticket
4.Make Payment
5.Exit
Enter Your Choice :
5
***Thanking You***
```



```
bat@matrix: ~/Desktop/Java /Day5
5
***Thanking You***
bat@matrix:~/Desktop/Java /Day5$ javac Busbooksys.java
bat@matrix:~/Desktop/Java /Day5$ java Busbooksys
Enter bus number :
TN 59 N 6067
Enter bus Route :
Madurai -> Chennai
Enter Bus Type :
Sleeper
Enter no of seats :
32
====Bus Booking System====
1.View Bus Detalis
2.Check Seat Avail
3.Book Ticket
4.Make Payment
5.Exit
Enter Your Choice :
1

Bus number : TN 59 N 6067
Route : Madurai -> Chennai
Bus Type : Sleeper
Available Seats : 32
====Bus Booking System====
1.View Bus Detalis
2.Check Seat Avail
3.Book Ticket
4.Make Payment
5.Exit
Enter Your Choice :
2
Available Seats : 32
====Bus Booking System====
1.View Bus Detalis
2.Check Seat Avail
3.Book Ticket
4.Make Payment
5.Exit
Enter Your Choice :
3
Enter Passenger name :
Shiva
Enter No.of seats to book :
3
Ticket booked
Seats updated : 29
====Bus Booking System====
1.View Bus Detalis
2.Check Seat Avail
```

```java
Busbooksys.java
~/Desktop/Java /Day5

import java.util.*;

interface Seatmgt {
  void showavail();
  void updateseats();
}

class Bus {
  protected String busno;
  protected String route;
  protected String bustype;
  protected int totalseats;

  public Bus(String busno, String route, String bustype, int totalseats) {
    this.busno = busno;
    this.route = route;
    this.bustype = bustype;
    this.totalseats = totalseats;
  }

  void showdetails() {
    System.out.println("\nBus number : "+busno+"\nRoute : "+route+"\nBus Type : "
+bustype+"\nAvailable Seats : "+totalseats);
  }
}

class Booking extends Bus {
  protected String passengername;
  protected int bookedseats;

  public Booking(String busno,String route, String bustype, int totalseats) {
    super(busno,route,bustype,totalseats);
  }

  void bookticket(String passengername, int seats) {
    if(seats <= totalseats) {
      this.passengername = passengername;
      this.bookedseats = seats;
      totalseats -= seats;
      System.out.println("Ticket booked");
    }
```

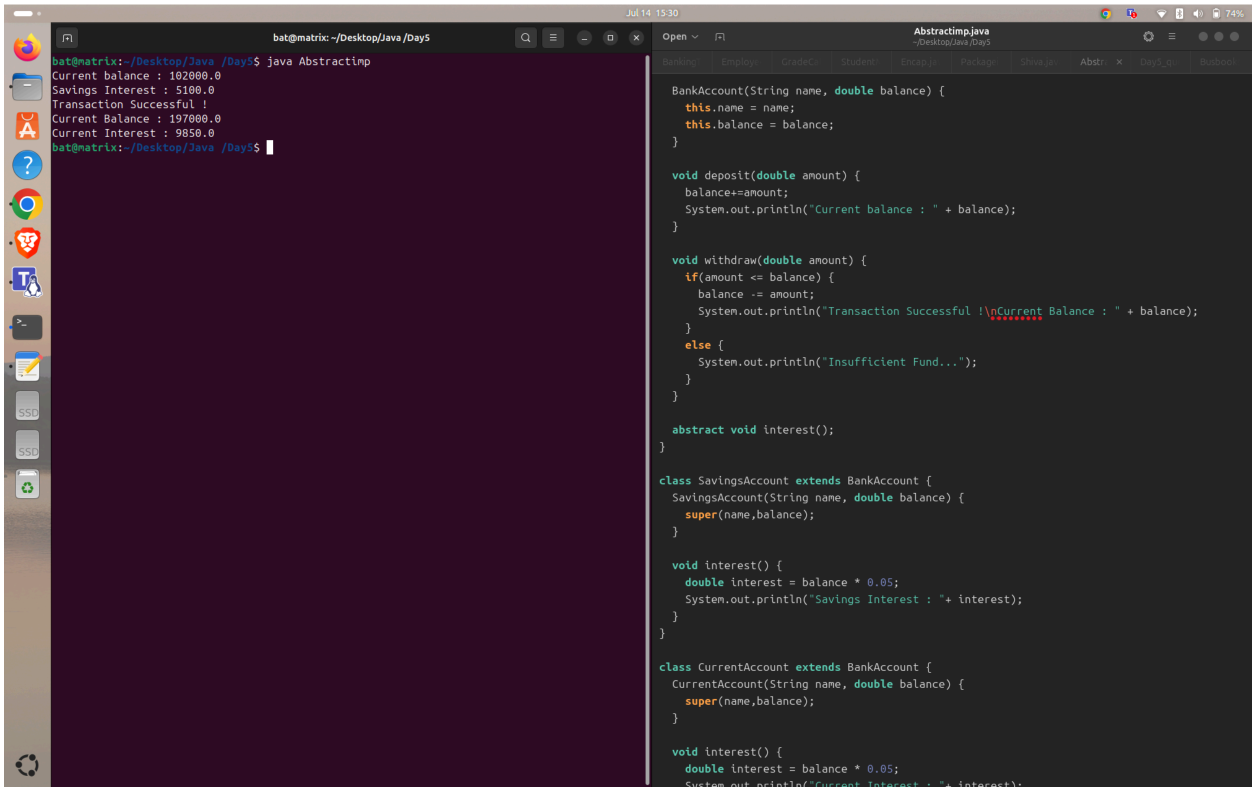
2.Abstraction implementation

```java
abstract class BankAccount {
	String name;
	double balance;

	BankAccount(String name, double balance) {
		this.name = name;
		this.balance = balance;
	}

	void deposit(double amount) {
		balance+=amount;
		System.out.println("Current balance : " + balance);
	}

	void withdraw(double amount) {
		if(amount <= balance) {
			balance -= amount;
			System.out.println("Transaction Successful !\nCurrent Balance : " +
balance);
		}
		else {
			System.out.println("Insufficient Fund...");
		}
	}

	abstract void interest();
}

class SavingsAccount extends BankAccount {
	SavingsAccount(String name, double balance) {
		super(name,balance);
	}

	void interest() {
		double interest = balance * 0.05;
		System.out.println("Savings Interest : "+ interest);
	}
}

class CurrentAccount extends BankAccount {
	CurrentAccount(String name, double balance) {
		super(name,balance);
	}

	void interest() {
		double interest = balance * 0.05;
		System.out.println("Current Interest : "+ interest);
	}
}
```

```
public class Abstractimp {
        public static void main(String[] args) {
                BankAccount shiva = new SavingsAccount("Shiva",100000);
                BankAccount balan = new CurrentAccount("Balan",200000);

                shiva.deposit(2000);
                shiva.interest();

                balan.withdraw(3000);
                balan.interest();
        }
}
```

Output:

```
Current balance : 102000.0
Savings Interest : 5100.0
Transaction Successful !
Current Balance : 197000.0
Current Interest : 9850.0
```



```
bat@matrix:~/Desktop/Java /Day5$ java Abstractimp
Current balance : 102000.0
Savings Interest : 5100.0
Transaction Successful !
Current Balance : 197000.0
Current Interest : 9850.0
bat@matrix:~/Desktop/Java /Day5$
```

```
  BankAccount(String name, double balance) {
    this.name = name;
    this.balance = balance;
  }

  void deposit(double amount) {
    balance+=amount;
    System.out.println("Current balance : " + balance);
  }

  void withdraw(double amount) {
    if(amount <= balance) {
      balance -= amount;
      System.out.println("Transaction Successful !\nCurrent Balance : " + balance);
    }
    else {
      System.out.println("Insufficient Fund...");
    }
  }

  abstract void interest();
}

class SavingsAccount extends BankAccount {
  SavingsAccount(String name, double balance) {
    super(name,balance);
  }

  void interest() {
    double interest = balance * 0.05;
    System.out.println("Savings Interest : "+ interest);
  }
}

class CurrentAccount extends BankAccount {
  CurrentAccount(String name, double balance) {
    super(name,balance);
  }

  void interest() {
    double interest = balance * 0.05;
    System.out.println("Current Interest : "+ interest);
```

3.Package Implementation

shivabalan/Shiva.java

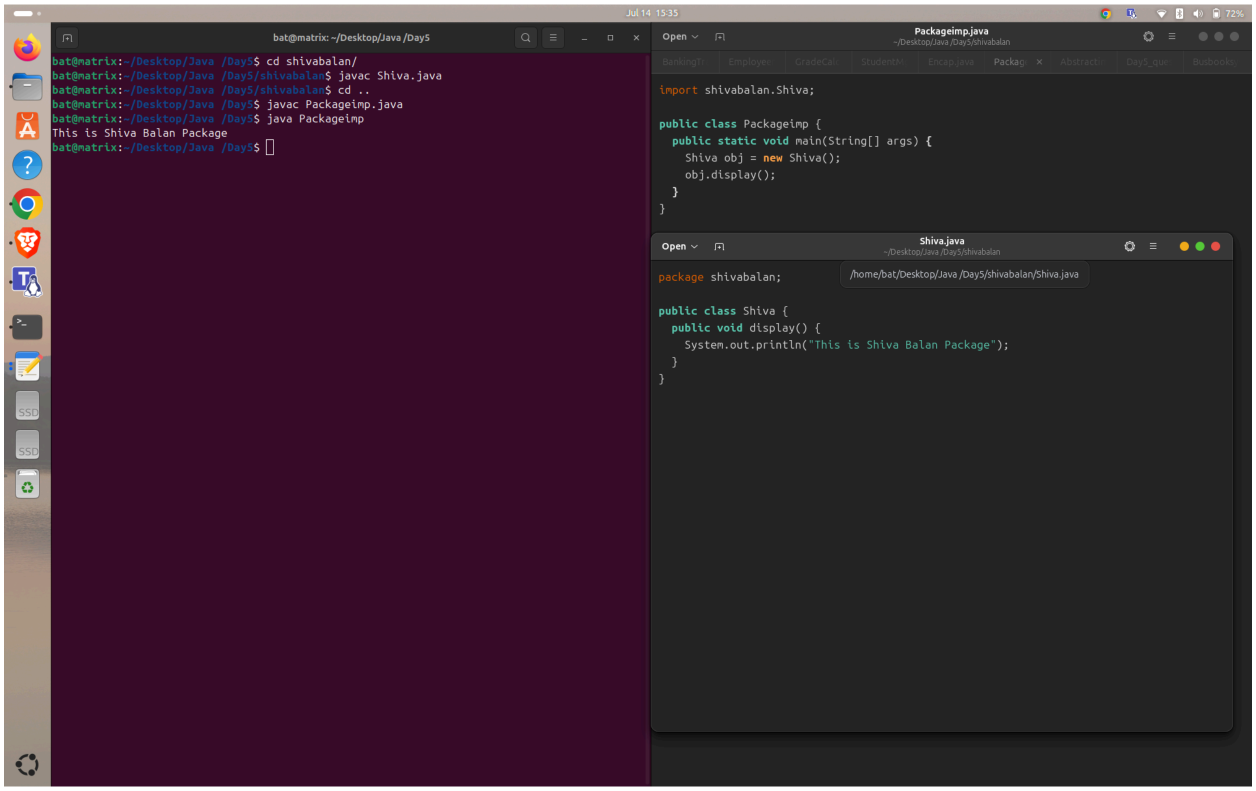
```java
package shivabalan;

public class Shiva {
	public void display() {
		System.out.println("This is Shiva Balan Package");
	}
}
```

Packageimp.java

```java
import shivabalan.Shiva;

public class Packageimp {
	public static void main(String[] args) {
		Shiva obj = new Shiva();
		obj.display();
	}
}
```

Output:

```
bat@matrix:~/Desktop/Java /Day5$ cd shivabalan/
bat@matrix:~/Desktop/Java /Day5/shivabalan$ javac Shiva.java
bat@matrix:~/Desktop/Java /Day5/shivabalan$ cd ..
bat@matrix:~/Desktop/Java /Day5$ javac Packageimp.java
bat@matrix:~/Desktop/Java /Day5$ java Packageimp
This is Shiva Balan Package
bat@matrix:~/Desktop/Java /Day5$
```


Jul 14 15:35
72%
bat@matrix: ~/Desktop/Java /Day5
Open
Packageimp.java
~/Desktop/Java /Day5/shivabalan
BankingTr
Employee
GradeCalc
StudentM
Encap.java
Package
Abstractin
Day5_que
Busbooks
Shiva.java
~/Desktop/Java /Day5/shivabalan
/home/bat/Desktop/Java /Day5/shivabalan/Shiva.java

```
bat@matrix:~/Desktop/Java /Day5$ cd shivabalan/
bat@matrix:~/Desktop/Java /Day5/shivabalan$ javac Shiva.java
bat@matrix:~/Desktop/Java /Day5/shivabalan$ cd ..
bat@matrix:~/Desktop/Java /Day5$ javac Packageimp.java
bat@matrix:~/Desktop/Java /Day5$ java Packageimp
This is Shiva Balan Package
bat@matrix:~/Desktop/Java /Day5$
```

```
import shivabalan.Shiva;

public class Packageimp {
  public static void main(String[] args) {
    Shiva obj = new Shiva();
    obj.display();
  }
}
```

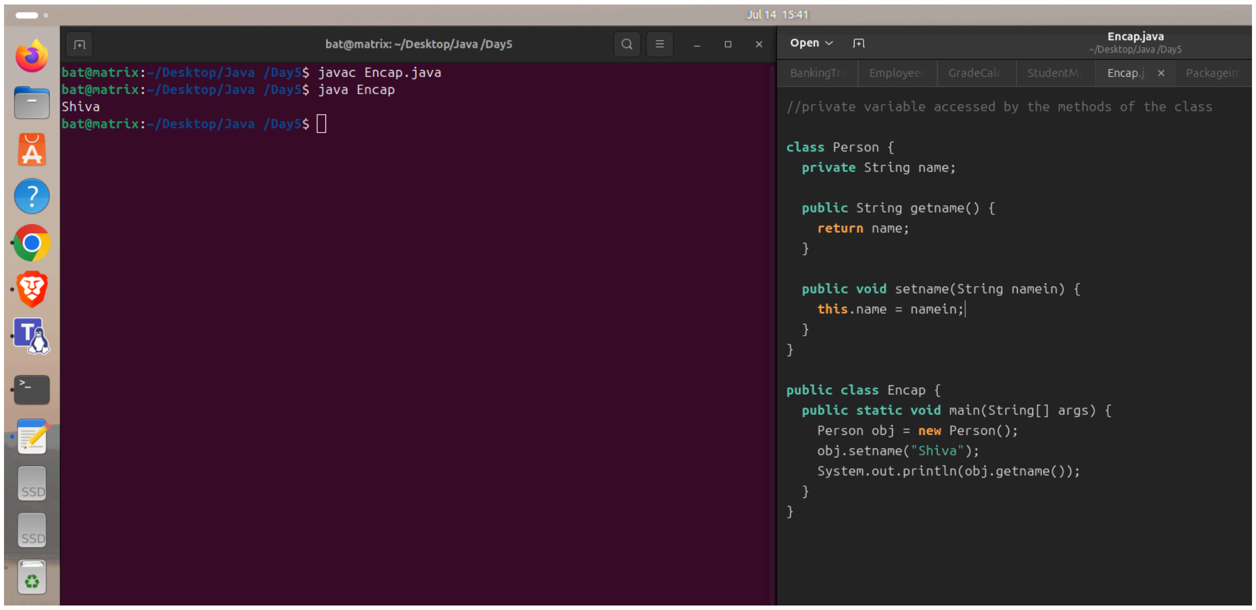
```
package shivabalan;

public class Shiva {
  public void display() {
    System.out.println("This is Shiva Balan Package");
  }
}
```

4.Encapsulation implementation

```
//private variable accessed by the methods of the class

class Person {
      private String name;

      public String getname() {
            return name;
      }

      public void setname(String namein) {
            this.name = namein;
      }
}

public class Encap {
      public static void main(String[] args) {
            Person obj = new Person();
            obj.setname("Shiva");
            System.out.println(obj.getname());
      }
}
```

Output:

```
bat@matrix:~/Desktop/Java /Day5$ javac Encap.java
bat@matrix:~/Desktop/Java /Day5$ java Encap
Shiva
```



```
bat@matrix:~/Desktop/Java /Day5$ javac Encap.java
bat@matrix:~/Desktop/Java /Day5$ java Encap
Shiva
bat@matrix:~/Desktop/Java /Day5$
```

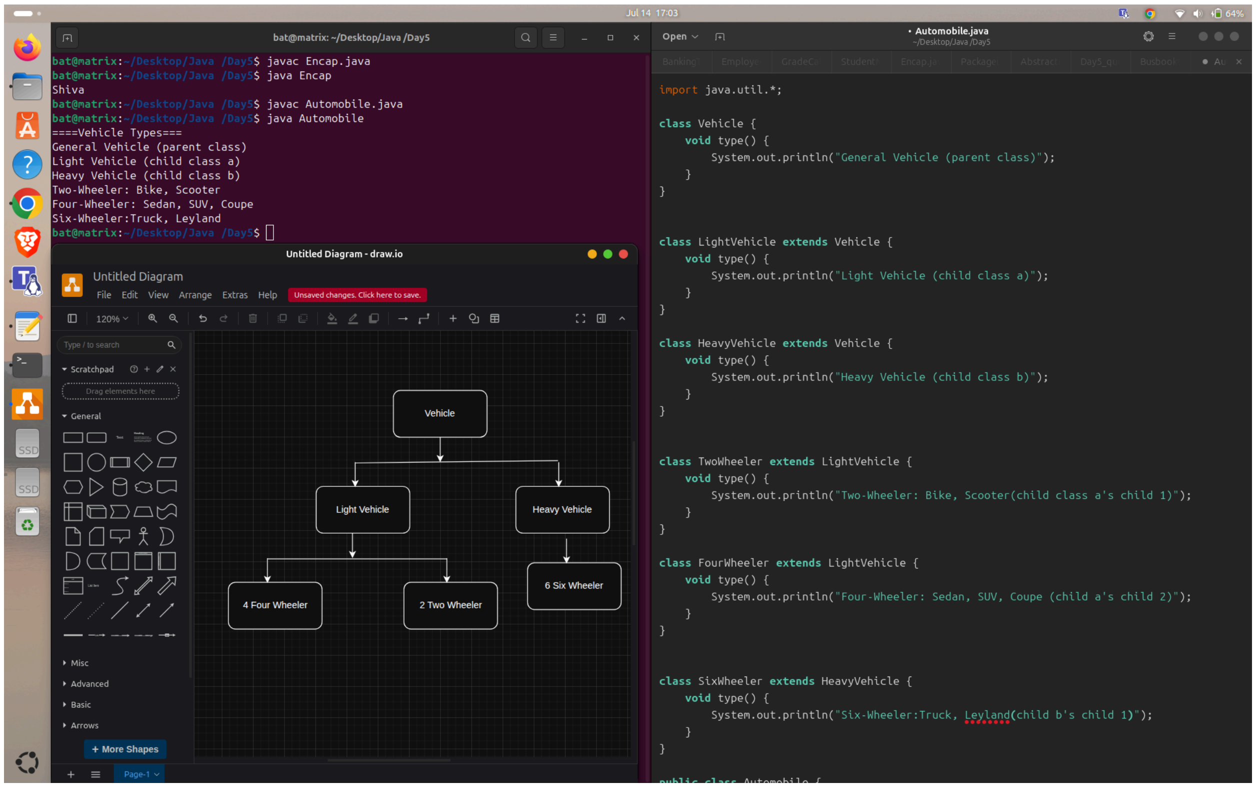
```
//private variable accessed by the methods of the class

class Person {
  private String name;

  public String getname() {
    return name;
  }

  public void setname(String namein) {
    this.name = namein;
  }
}

public class Encap {
  public static void main(String[] args) {
    Person obj = new Person();
    obj.setname("Shiva");
    System.out.println(obj.getname());
  }
}
```

5.Multilevel and hierarchical inheritance

```
import java.util.*;

class Vehicle {
    void type() {
        System.out.println("General Vehicle (parent class)");
    }
}


class LightVehicle extends Vehicle {
    void type() {
        System.out.println("Light Vehicle (child class a)");
    }
}

class HeavyVehicle extends Vehicle {
    void type() {
        System.out.println("Heavy Vehicle (child class b)");
    }
}


class TwoWheeler extends LightVehicle {
    void type() {
        System.out.println("Two-Wheeler: Bike, Scooter(child class
a's child 1)");
    }
}

class FourWheeler extends LightVehicle {
    void type() {
        System.out.println("Four-Wheeler: Sedan, SUV, Coupe (child
a's child 2)");
    }
}


class SixWheeler extends HeavyVehicle {
    void type() {
        System.out.println("Six-Wheeler:Truck, Leyland(child b's
child 1)");
    }
}

public class Automobile {
    public static void main(String[] args) {
        Vehicle general = new Vehicle();
        Vehicle light = new LightVehicle();
        Vehicle heavy = new HeavyVehicle();
        Vehicle bike = new TwoWheeler();
        Vehicle car = new FourWheeler();
        Vehicle truck = new SixWheeler();


        System.out.println("====Vehicle Types===");
        general.type();
        light.type();
        heavy.type();
        bike.type();
        car.type();
        truck.type();
    }
}
```

Output:

```
====Vehicle Types===
General Vehicle (parent class)
Light Vehicle (child class a)
Heavy Vehicle (child class b)
Two-Wheeler: Bike, Scooter
Four-Wheeler: Sedan, SUV, Coupe
Six-Wheeler:Truck, Leyland
```



```
bat@matrix:~/Desktop/Java /Day5$ javac Encap.java
bat@matrix:~/Desktop/Java /Day5$ java Encap
Shiva
bat@matrix:~/Desktop/Java /Day5$ javac Automobile.java
bat@matrix:~/Desktop/Java /Day5$ java Automobile
====Vehicle Types===
General Vehicle (parent class)
Light Vehicle (child class a)
Heavy Vehicle (child class b)
Two-Wheeler: Bike, Scooter
Four-Wheeler: Sedan, SUV, Coupe
Six-Wheeler:Truck, Leyland
bat@matrix:~/Desktop/Java /Day5$ 
```

```java
import java.util.*;

class Vehicle {
    void type() {
        System.out.println("General Vehicle (parent class)");
    }
}


class LightVehicle extends Vehicle {
    void type() {
        System.out.println("Light Vehicle (child class a)");
    }
}

class HeavyVehicle extends Vehicle {
    void type() {
        System.out.println("Heavy Vehicle (child class b)");
    }
}


class TwoWheeler extends LightVehicle {
    void type() {
        System.out.println("Two-Wheeler: Bike, Scooter(child class a's child 1)");
    }
}

class FourWheeler extends LightVehicle {
    void type() {
        System.out.println("Four-Wheeler: Sedan, SUV, Coupe (child a's child 2)");
    }
}


class SixWheeler extends HeavyVehicle {
    void type() {
        System.out.println("Six-Wheeler:Truck, Leyland(child b's child 1)");
    }
}

public class Automobile {
```