|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module** | **I-Practice (Mandatory)** | **Solution** | **I-Practice (Bridges)** | **Solution** | **I-Practice (Hots)** | **Solution** | Remarks |
| Basic elements of java | **Customized Welcome Message** | import java.util.Scanner;  class Main {  public static void main(String[] args) {  String name;  Scanner sc = new Scanner(System.in);  do {  System.out.println("Enter your name");  name = sc.nextLine();  if(name.length()>50)  System.out.println("Name should be less than 50 characters!");  }while(name.length()>50);  sc.close();  System.out.println("Hello "+name+" ! Welcome to Amphi Event Management System");  }  } | WELCOME MESSAGE | public class Main{  public static void main(String[] args){  System.out.print("Welcome to Amphi Event Management System");  }  } | **Pranav and Change** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int amount, count=0, rem;  amount= sc.nextInt();  sc.close();  count+=amount/100;  rem = amount%100;  count+=rem/50;  rem = rem%50;  count+=rem/10;  rem = rem%10;  count+=rem/5;  rem = rem%5;  count+=rem/2;  rem = rem%2;  count+=rem;    System.out.println(count);  }  } |  |
| **TOTAL EXPENSES FOR THE EVENT** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  int branding, travel, food, logistics;  double total, brPercentage, trPercentage, foPercentage, loPercentage;  Scanner sc = new Scanner(System.in);  System.out.println("Enter branding expenses");  branding = sc.nextInt();  System.out.println("Enter travel expenses");  travel = sc.nextInt();  System.out.println("Enter food expenses");  food = sc.nextInt();  System.out.println("Enter logistics expenses");  logistics = sc.nextInt();  sc.close();  total = branding+travel+food+logistics;  brPercentage = (branding/total)\*100;  trPercentage = (travel/total)\*100;  foPercentage = (food/total)\*100;  loPercentage = (logistics/total)\*100;  System.out.printf("Total expenses : Rs.%.2f\nBranding expenses percentage : %.2f%%\nTravel expenses percentage : %.2f%%\nFood expenses percentage : %.2f%%\nLogistics expenses percentage : %.2f%%", total,brPercentage,trPercentage,foPercentage,loPercentage);  }  } | **Ticket type** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  int age;  Scanner sc = new Scanner(System.in);  age = sc.nextInt();  if(age<15)  System.out.println("Child Ticket");  else  System.out.println("Adult Ticket");  sc.close();  }  } | **Series 2** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num, count=0, i=20, j=0;  num = sc.nextInt();    while(count<num) {  System.out.print(i+" ");  i+=40+j;  count++;  j=j+4;  }  sc.close();  }  } |  |
| **Thrill ride** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  int age;  Scanner sc = new Scanner(System.in);  age = sc.nextInt();  if(age<15 || age>60)  System.out.println("Not Allowed");  else  System.out.println("Allowed");  sc.close();  }  } | S3P3-LUCKY PAIRS |  | AF\_PATTERN5 |  |  |
| CHARACTER PATTERN 3 | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  int num;  Scanner sc = new Scanner(System.in);  num = sc.nextInt();  for(int i=0; i<num; i++) {  for(int j=0; j<i+1; j++) {  System.out.print("\*");  }  System.out.println();  }  sc.close();  }  } | **Character Pattern 1A** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num = sc.nextInt();    for(int i=0; i<num; i++) {  for(int j=0; j<num; j++) {  System.out.print("\*");  }  System.out.println();  }  sc.close();  }  } | PATTERN 1 |  |  |
| **Series 1** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int nth = sc.nextInt();  int num, count, i;  num=1;  count=0;  while (count < nth){  num=num+1;  for (i = 2; i <= num; i++){  if (num % i == 0) {  break;  }  }  if ( i == num){  System.out.print(num+" ");  count = count+1;  }  }  }  } | **Character Pattern 1B** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num,i=0,j=0;  num= sc.nextInt();  while(i<num) {  while(j<num) {  System.out.print("\*");  j++;  }  i++;  j=0;  System.out.println();  }  sc.close();  }  } | PATTERN 2 |  |  |
|  |  | **Alphabet Pattern 1** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  int num;  Scanner sc = new Scanner(System.in);  num = sc.nextInt();  for(int i=0; i<num; i++) {  for(int j=65; j<=65+i; j++) {  System.out.print((char)j);  }  System.out.println();  }  sc.close();  }  } | PATTERN 3 |  |  |
|  |  |  | **Alphabet Pattern 4** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  int num;  Scanner sc = new Scanner(System.in);  num = sc.nextInt();  for(int i=64+num; i>=65; i--) {  for(int j=64+num; j>=i; j--) {  System.out.print((char)j);  }  System.out.println();  }  sc.close();  }  } | DAY 1 CRITERIA | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int N, num = 0;  N = sc.nextInt();  for (int i = 1; i <= N; i++) {  if(i%2==0)  num=(2\*i\*i)+1;  else  num=(2\*i\*i)-1;  System.out.print(num+ " ");  }  sc.close();  }  } |  |
|  |  |  | **Day 11 Criteria** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  int num;  Scanner sc = new Scanner(System.in);  num = sc.nextInt();  for(int i=2; i<num+2; i++) {  System.out.print((i\*(i+1)\*(i+2))+" ");  }  }  } | DAY 2 CRITERIA | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num = sc.nextInt();  for(int i=1; i<=num; i++) {  System.out.print(((i\*i)+(3\*i))/2+" ");  }  sc.close();  }  } |  |
|  |  |  | **Day 13 Criteria** | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num = sc.nextInt();  for(int i=2; i<=num\*2; i+=2) {  System.out.print((i\*i)+1+" ");  }  sc.close();  }  } | DAY 3 CRITERIA | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int num, count=0;  num= sc.nextInt();  int i=5;  while(count<num) {  i+=4\*count;  System.out.print(i+" ");  count++;  }  sc.close();  }  } |  |
| OOPS, Classes & Methods | DISPLAY ITEM TYPE | import java.io.\*;  import java.util.Scanner;  class Main{  public static void main(String[] args) throws Exception{  Scanner sc = new Scanner(System.in);  ItemType it = new ItemType();  System.out.println("Enter the item type name");  String name=sc.nextLine();  System.out.println("Enter the cost per day");  Double costPerDay = sc.nextDouble();  System.out.println("Enter the deposit");  Double deposit = sc.nextDouble();  it.setName(name);  it.setCostPerDay(costPerDay);  it.setDeposit(deposit);  sc.close();    it.display();  }  }  import java.text.\*;  public class ItemType {  private String name;  private double costPerDay;  private double deposit;  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public double getCostPerDay() {  return costPerDay;  }  public void setCostPerDay(double costPerDay) {  this.costPerDay = costPerDay;  }  public double getDeposit() {  return deposit;  }  public void setDeposit(double deposit) {  this.deposit = deposit;  }  public void display(){  System.out.println("Item type details");  System.out.printf("Name : %s\nCostPerDay : %.2f\nDeposit : %.2f",name,costPerDay,deposit);  }  } | CUSTOMER CLASS WITH CONSTRUCTOR | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("Enter the Customer Details\nEnter the name");  String name = sc.nextLine();  System.out.println("Enter the email");  String email = sc.nextLine();  System.out.println("Enter the type");  String type = sc.nextLine();  System.out.println("Enter the location");  String location = sc.nextLine();    Customer c = new Customer(name, email, type, location);  c.displayDetails();  }  }  public class Customer {  public String customerName;  public String customerEmail;  public String customerType;  public String customerAddress;    public Customer(String customerName, String customerEmail, String customerType, String customerAddress) {  super();  this.customerName = customerName;  this.customerEmail = customerEmail;  this.customerType = customerType;  this.customerAddress = customerAddress;  }  public void displayDetails() {  System.out.println("Name: " + customerName + "\nE-mail: " + customerEmail + "\nType: "  + customerType + "\nLocation: " + customerAddress);  }  } | CUSTOMER ADDRESS | import java.util.\*;  public class Main{  public static void main(String[] args){  Scanner sc = new Scanner(System.in);  String street;  String city;  int pincode;  String country;    System.out.println("Enter Customer Address");  System.out.println("Enter the street");  street = sc.nextLine();    System.out.println("Enter the city");  city = sc.nextLine();    System.out.println("Enter the pincode");  pincode = sc.nextInt();  sc.nextLine();    System.out.println("Enter the country");  country = sc.nextLine();    Address a = new Address(street, city, pincode, country);  a.displayAddress();  }  }  public class Address{    String street;  String city;  int pincode;  String country;    public Address(String street, String city, int pincode, String country) {  super();  this.street = street;  this.city = city;  this.pincode = pincode;  this.country = country;  }    void displayAddress() {  System.out.println("Street: "+ street);  System.out.println("City: "+ city);  System.out.println("Pincode: "+ pincode);  System.out.println("Country: "+ country);  }  } |  |
|  | COMPARE PHONE NUMBER - JAVA | import java.io.BufferedReader;  import java.io.InputStreamReader;  import java.io.\*;  class Main{  public static void main(String[] args) throws Exception{    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));    System.out.println("Enter Name");  String name1 = br.readLine();    System.out.println("Enter UserName");  String username1 = br.readLine();    System.out.println("Enter Password");  String password1 = br.readLine();    System.out.println("Enter PhoneNo");  long phoneNo1 = Long.parseLong(br.readLine());    User user1 = new User(name1,username1,password1,phoneNo1);  System.out.println("Enter Name");  String name2 = br.readLine();    System.out.println("Enter UserName");  String username2 = br.readLine();    System.out.println("Enter Password");  String password2 = br.readLine();    System.out.println("Enter PhoneNo");  long phoneNo2 = Long.parseLong(br.readLine());    User user2 = new User(name2,username2,password2,phoneNo2);    if(user1.comparePhoneNumber(user2)){  System.out.println("Same Users");  }  else  {  System.out.println("Different Users");  }  }  }  public class User {  private String name;  private String userName;  private String password;  private long phoneNo;    public User(){}  public User(String name,String userName,String password,long phoneNo) {    this.name = name;  this.userName = userName;  this.password = password;  this.phoneNo = phoneNo;  }    public String getName() {  return name;  }  public String getUserName() {  return userName;  }    public String getPassword() {  return password;  }    public long getPhoneNo() {  return phoneNo;  }    public void setName(String name) {  this.name = name;  }    public void setUserName(String userName) {  this.userName = userName;  }    public void setPassword(String password) {  this.password = password;  }    public void setPhoneNo(long phoneNo) {  this.phoneNo = phoneNo;  }    public boolean comparePhoneNumber(User user) {  return this.phoneNo == user.phoneNo;  }  } | PAYMENT-METHOD OVERLOADING - JAVA | import java.util.Scanner;  import java.io.BufferedReader;  import java.io.IOException;  import java.io.InputStreamReader;  import java.util.Scanner;  public class Main {  public static void main(String[] args) throws IOException {  int ch;  double amount;  BufferedReader reader =new BufferedReader(new InputStreamReader(System.in));  Scanner sc = new Scanner(System.in);  System.out.println("Enter the mode of Payment:\n1.Cash Payment\n2.Wallet Payment\n3.Credit Card");  ch = sc.nextInt();  TicketBooking tb = new TicketBooking();  switch(ch) {  case 1:  System.out.println("Enter the Amount of Payment:");  amount = sc.nextDouble();  tb.makePayment(amount);  break;  case 2:  System.out.println("Enter the Wallet Number:");  String walletNo = sc.next();  System.out.println("Enter the Amount of Payment:");  amount = sc.nextDouble();  tb.makePayment(walletNo, amount);  break;  case 3:  System.out.println("Enter the Credit Card Number:");  String ccNo = reader.readLine();  System.out.println("Enter the Validity Date(dd/MM/yyyy):");  String ccv = reader.readLine();  System.out.println("Enter the Card Holder Name:");  String name = reader.readLine();  System.out.println("Enter the Amount of Payment:");  amount = Double.parseDouble(reader.readLine());  tb.makePayment(ccNo, ccv, name, amount);  break;  default:  System.out.println("Please select the correct mode of payment...");  }  }  }  public class TicketBooking {  public void makePayment(double amount) {  System.out.println("You have selected the Cash payment mode\nThe Amount is Rs."+(int)amount);  }  public void makePayment(String walletNumber, double amount) {  System.out.println("You have selected the Wallet payment mode\nWallet Number: "+walletNumber+"\nThe Amount is Rs."+(int)amount);  }  public void makePayment(String creditCard, String ccv, String name, double amount) {  System.out.println("You have selected the Credit Card payment mode\nCreditCard Number: "+creditCard+"\nValidity Date: "+ccv+"\nCard Holder Name: "+name+"\nThe Amount is Rs."+(int)amount);  }  } | F1P5 - BEST MOBILE PLAN |  |  |
|  | RECTANGLE DIMENSION CHANGE - INSTANCEOF OPERATOR | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("Enter the length of the rectangle");  int l = sc.nextInt();  System.out.println("Enter the width of the rectangle");  int w = sc.nextInt();  Rectangle r = new Rectangle(l, w);  r.display();  System.out.println("Area of the Rectangle:"+r.area());  System.out.println("Enter the new dimension");  int fact = sc.nextInt();  Rectangle r2 = r.dimensionChange(fact);  if(r2 instanceof Rectangle) {  r2.display();  System.out.println("Area of the Rectangle:"+r2.area());  }  sc.close();  }  }  public class Rectangle {  private int length;  private int width;  public int getLength() {  return length;  }  public void setLength(int length) {  this.length = length;  }  public int getWidth() {  return width;  }  public void setWidth(int width) {  this.width = width;  }  public Rectangle(int length, int width) {  super();  this.length = length;  this.width = width;  }    int area() {  return length\*width;  }    void display() {  System.out.println("Rectangle Dimension\nLength:"+length+"\nWidth:"+width);  }    Rectangle dimensionChange(int d) {  return new Rectangle(length\*d,width\*d);  }  } | SUM OF AN ARRAY | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("Enter n :");  int n = sc.nextInt();  int sum=0;  int arr[] = new int[n];  for(int i=0; i<n; i++) {  arr[i] = sc.nextInt();  }  for(int i:arr) {  sum= sum+i;  }    System.out.println("Sum of array elements is : "+sum);    sc.close();  }  } | COMMAND LINE ARGUMENTS II | public class Main{  public static void main(String[] args){  int a, b;  if(args.length == 2) {  a = Integer.parseInt(args[0]);  b = Integer.parseInt(args[1]);    System.out.println("The sum of "+a+" and "+b+" is "+(a+b));  }  else  System.out.println("Invalid Input");  }  } |  |
|  | F2P4 - SIMPLIFIED FRACTION | import java.util.\*;  public class Main{  public static void main(String args[]){  Scanner sc = new Scanner(System.in);  int num = sc.nextInt();  int den = sc.nextInt();    Fraction f = new Fraction();  f.printValue(num, den);    sc.close();  }  }  public class Fraction {  void printValue(int num, int den) {  if(num==0){  System.out.println(num);  }  else if(num/den==0) {  System.out.println(asFraction(num,den));  }  else if(num%den==0) {  System.out.println(num/den);  }  else {  int mod = num%den;  System.out.println(num/den+" "+asFraction(mod,den));  }  }  public int gcd(int a, int b) {  return b == 0 ? a : gcd(b, a % b);  }  public String asFraction(int a, int b) {  int gcd = gcd(a, b);  return (a / gcd) + "/" + (b / gcd);  }  } | COMMAND LINE ARGUMENT - PRINT STRING | class Main {  public static void main(String[] args) {  System.out.println(args[0]+" - Command Line Arguments");  }  } |  |  |  |
|  | ARRAY - FOR EACH LOOP | import java.io.\*;  import java.util.\*;  public class Main{  public static void main (String[] args) throws Exception{  Scanner sc = new Scanner(System.in);  System.out.println("Enter n :");  int n = sc.nextInt();  int num[] = new int[n];  String str[] = new String[n];    System.out.println("Enter numbers : ");  for(int i=0;i<n;i++) {  num[i] = sc.nextInt();  }  System.out.println("Enter strings : ");  for(int i=0;i<n;i++) {  str[i] = sc.next();  }    System.out.println("Displaying numbers");  for(int i=0;i<n;i++) {  System.out.println(num[i]);  }    System.out.println("Displaying strings");  for(int i=0;i<n;i++) {  System.out.println(str[i]);  }  sc.close();  }  } |  |  |  |  |  |
|  | COMMAND LINE ARGUMENT - COUNT | public class Main{  public static void main(String[] args){  System.out.println("Arguments :");  for(int n=0;n<args.length;n++) {  System.out.println(args[n]);  }  System.out.println("Number of arguments is "+args.length);  }  } |  |  |  |  |  |
| Inheritance | SINGLE INHERITANCE - HCL | import java.util.Scanner;  public class Main {  public static void main(String[] args){    Scanner sc=new Scanner(System.in);      System.out.println("Enter the name :");  String name = sc.nextLine();  System.out.println("Enter Date of Birth :");  String dob = sc.nextLine();  System.out.println("Enter Gender :");  String gender = sc.nextLine();    System.out.println("Enter Mobile Number :");  String mobile = sc.nextLine();    System.out.println("Enter Blood Group :");  String bloodGroup = sc.nextLine();    System.out.println("Enter Blood Bank Name :");  String bbank = sc.nextLine();    System.out.println("Enter Donor Type :");  String dtype = sc.nextLine();  System.out.println("Enter Donation Date :");  String ddate = sc.nextLine();    Donor d = new Donor(name,dob,gender,mobile,bloodGroup,bbank,dtype,ddate);  d.displayDonationDetails();    }  }  class Donor extends Person{    private String bloodBankName;  private String donorType;  private String donationDate;    public Donor(String name, String dateOfBirth, String gender, String mobileNumber, String bloodGroup,String bloodBankName, String donorType, String donationDate) {  super(name, dateOfBirth, gender, mobileNumber, bloodGroup);  this.bloodBankName = bloodBankName;  this.donorType = donorType;  this.donationDate = donationDate;  }  public String getBloodBankName() {  return bloodBankName;  }  public void setBloodBankName(String bloodBankName) {  this.bloodBankName = bloodBankName;  }  public String getDonorType() {  return donorType;  }  public void setDonorType(String donorType) {  this.donorType = donorType;  }  public String getDonationDate() {  return donationDate;  }  public void setDonationDate(String donationDate) {  this.donationDate = donationDate;  }    public void displayDonationDetails( ) {  System.out.println("Donation Details :");  super.displayDetails();  System.out.println("Blood Bank Name : " + bloodBankName);  System.out.println("Donor Type : " + donorType);  System.out.println("Donation Date : " + donationDate);  }    }  class Person{    private String name;  private String dateOfBirth;  private String gender;  private String mobileNumber;  private String bloodGroup;    public Person(String name, String dateOfBirth, String gender, String mobileNumber, String bloodGroup)  {  this.name = name;  this.dateOfBirth = dateOfBirth;  this.gender = gender;  this.mobileNumber = mobileNumber;  this.bloodGroup = bloodGroup;  }    public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public String getDateOfBirth() {  return dateOfBirth;  }  public void setDateOfBirth(String dateOfBirth) {  this.dateOfBirth = dateOfBirth;  }  public String getGender() {  return gender;  }  public void setGender(String gender) {  this.gender = gender;  }  public String getMobileNumber() {  return mobileNumber;  }  public void setMobileNumber(String mobileNumber) {  this.mobileNumber = mobileNumber;  }  public String getBloodGroup() {  return bloodGroup;  }  public void setBloodGroup(String bloodGroup) {  this.bloodGroup = bloodGroup;  }    public void displayDetails() {  System.out.println("Name : " + name);  System.out.println("Date Of Birth : " + dateOfBirth);  System.out.println("Gender : " + gender);  System.out.println("Mobile Number : " + mobileNumber);  System.out.println("Blood Group : " + bloodGroup);  }  } | EVENT DETAILS -SIMPLE OVERRIDE | import java.util.Scanner;  public class Main{  public static void main(String[] args){  Scanner scan = new Scanner(System.in);  System.out.println("Enter Event Name");  String name = scan.nextLine();    System.out.println("Enter Detail");  String detail = scan.nextLine();  System.out.println("Enter Organizer");  String organizer = scan.nextLine();    System.out.println("Select Event Type 1.Exhibition 2.StageEvent");  int option = scan.nextInt();    if (option == 1) {    System.out.println("Enter stall count");  int stall = scan.nextInt();    Exhibition ex = new Exhibition(name,detail,organizer,stall);  System.out.println(ex.toString());  ex.totalCredit();  }  else if (option == 2) {    System.out.println("Enter Total shows");  int shows = scan.nextInt();    System.out.println("Enter seats per show");  int show\_s = scan.nextInt();    Event se = new StageEvent(name,detail,organizer,shows,show\_s);  System.out.println(se.toString());  se.totalCredit();  }    else {  System.out.println("Enter valid choice");  }  }  }  public class Exhibition extends Event {  private int stallCount;  public Exhibition(String name, String detail, String organizer, int stallCount) {  super(name, detail, organizer);  this.stallCount = stallCount;  }  public int getStallCount() {  return stallCount;  }  public void setStallCount(int stallCount) {  this.stallCount = stallCount;  }  @Override  public void totalCredit() {  super.totalCredit();  System.out.println("Total Credit Gained is "+stallCount\*50);  }    @Override  public String toString() {  System.out.print(super.toString());  return "\nStall Count : " + stallCount;  }  }  class Event{  private String name;  private String detail;  private String organizer;  public Event(String name, String detail, String organizer) {  super();  this.name = name;  this.detail = detail;  this.organizer = organizer;  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public String getDetail() {  return detail;  }  public void setDetail(String detail) {  this.detail = detail;  }  public String getOrganizer() {  return organizer;  }  public void setOrganizer(String organizer) {  this.organizer = organizer;  }  @Override  public String toString() {  return "Event Name : " + name + "\nEvent Detail : " + detail + "\nEvent Organizer : " + organizer;  }  public void totalCredit() {  System.out.println("Credit Details");  }  }  class StageEvent extends Event{  private int totalShow;  private int seatPerShow;  public StageEvent(String name, String detail, String organizer,int totalShow, int seatPerShow ) {  super(name, detail, organizer);  this.totalShow = totalShow;  this.seatPerShow = seatPerShow;  }  public int getTotalShow() {  return totalShow;  }  public void setTotalShow(int totalShow) {  this.totalShow = totalShow;  }  public int getSeatPerShow() {  return seatPerShow;  }  public void setSeatPerShow(int seatPerShow) {  this.seatPerShow = seatPerShow;  }  @Override  public void totalCredit() {  super.totalCredit();  System.out.println("Total Credit Gained is " + ((totalShow\* seatPerShow )\* 100));  }  @Override  public String toString() {  System.out.print(super.toString());  return "\nTotal Events : " + totalShow + "\nTotal Seats : " + seatPerShow;  }  } | ACCOUNT DETAILS - HCL | import java.util.\*;  public class Main {  public static void main(String[] args){  // TODO Auto-generated method stub  Scanner sc = new Scanner(System.in);  String input1, holderName, iFSCCode, accType, orgName;  double interestRate;  long accNumber, contactNumber, TIN;    System.out.println("Enter User Details(HolderName,Account Number,IFSC code,Contact Number)");  input1 = sc.nextLine();    System.out.println("Enter Account Type");  accType = sc.nextLine();      if(accType.equals("saving")) {  System.out.println("Enter Interest Rate");  interestRate = sc.nextDouble();    String[] sArray = input1.split(",");  holderName = sArray[0];  accNumber = Long.parseLong(sArray[1]);  iFSCCode = sArray[2];  contactNumber = Long.parseLong(sArray[3]);    SavingAccount sa = new SavingAccount(holderName,accNumber,iFSCCode,contactNumber,interestRate);  sa.display();  }  else if(accType.equals("current")) {  System.out.println("Enter organization Name");  orgName = sc.nextLine();    System.out.println("Enter TIN number");  TIN = sc.nextLong();    String[] sArray = input1.split(",");  holderName = sArray[0];  accNumber = Long.parseLong(sArray[1]);  iFSCCode = sArray[2];  contactNumber = Long.parseLong(sArray[3]);    CurrentAccount ca = new CurrentAccount(holderName,accNumber,iFSCCode,contactNumber,orgName,TIN);  ca.display();  }  else {  System.out.println("Enter valid Account Type");  System.exit(0);  }  }  }  class Account{    private String holderName;  private long accNumber;  private String IFSCCode;  private long contactNumber;      public Account(String holderName, long accNumber, String iFSCCode, long contactNumber) {  super();  this.holderName = holderName;  this.accNumber = accNumber;  IFSCCode = iFSCCode;  this.contactNumber = contactNumber;  }  public void display() {  System.out.println("Your Contact Details");  System.out.println("HolderName : "+ holderName);  System.out.println("Account Number : "+ accNumber);  System.out.println("IFSCCode : "+ IFSCCode);  System.out.println("ContactNumber : "+contactNumber);  }  }  class SavingAccount extends Account{  private double interestRate;    public SavingAccount(String holderName, long accNumber, String iFSCCode, long contactNumber, double interestRate) {  super(holderName, accNumber, iFSCCode, contactNumber);  // TODO Auto-generated constructor stub  this.interestRate = interestRate;  }  public void display( ){  super.display();  if(interestRate % 1 == 0){  int x = (int)Math.round(interestRate);  System.out.printf("Interest Rate : "+ x);  }  else  System.out.printf("Interest Rate : "+ interestRate);  }  }  public class CurrentAccount extends Account{  private String orgName;  private long TIN;    public CurrentAccount(String holderName, long accNumber, String iFSCCode, long contactNumber, String orgName,long TIN) {  super(holderName, accNumber, iFSCCode, contactNumber);  this.orgName = orgName;  this.TIN = TIN;  }  public void display( ){  super.display();  System.out.println("Organization Name : "+ orgName);  System.out.println("TIN : "+ TIN);  }  } |  |
|  | CALCULATE REWARD POINTS - HCL | import java.io.\*;  import java.util.Scanner;  public class Main {  public static void main(String[] args){  Scanner sc = new Scanner(System.in);  VISACard card;  String response;  do{  System.out.println("Enter the transaction detail");  String details = sc.nextLine();    String[] detailArr = details.split(",");    String type = detailArr[0];  double amount = Double.parseDouble(detailArr[1]);  String cardType = detailArr[2];  switch(cardType){  case "VISA card":  card = new VISACard();  System.out.format("Total reward points earned in this transaction is %.2f%n", card.computeRewardPoints(type,amount));  break;  case "HPVISA card":  card = new HPVISACard();  System.out.format("Total reward points earned in this transaction is %.2f%n", card.computeRewardPoints(type,amount));  break;    default :  System.out.println("Invalid data");  break;  }  System.out.println("Do you want to continue?(Yes/No)");  response = sc.nextLine();  }while(response.equals("Yes"));  }  }  class HPVISACard extends VISACard{  public double computeRewardPoints(String type, double amount){  double reward;  if(type.equals("Fuel")){  reward = super.computeRewardPoints(type,amount) + (double)10;  }else{  reward = super.computeRewardPoints(type,amount);  }  return reward;  }  }  public class VISACard {    public double computeRewardPoints(String type, double amount){  double reward = amount\* 0.01;  return reward;  }  } | STUDENT-FEEDBACK(SINGLE INHERITANCE) | import java.io.\*;  import java.util.\*;  public class Main{  public static void main(String[] args) throws Exception, IOException{  Scanner scan = new Scanner(System.in);  System.out.println("Enter the student id");  int id = scan.nextInt();    System.out.println("Enter the student name");  scan.nextLine();  String name = scan.nextLine();  System.out.println("Enter the department");  String department = scan.nextLine();    System.out.println("Enter the course id");  int c\_id = scan.nextInt();    System.out.println("Enter the Rating id");  int r\_id = scan.nextInt();    System.out.println("Enter review");  scan.nextLine();  String review = scan.nextLine();    System.out.println("Enter number of stars");  int stars = scan.nextInt();    Student s1 = new StudentRating(id,name,department,c\_id,r\_id,review,stars);  System.out.println(s1.toString());  scan.close();  }  }  class StudentRating extends Student{  private int id;  private String review;  private int stars;  private int rating;  public StudentRating(int id, String name, String department, int courseId,int id2, String review, int stars) {  super(id, name, department, courseId);  this.id = id2;  this.review = review;  this.stars = stars;  }  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }  public String getReview() {  return review;  }  public void setReview(String review) {  this.review = review;  }  public int getStars() {  return stars;  }  public void setStars(int stars) {  this.stars = stars;  }  public int getRating() {  return rating;  }  public void setRating(int rating) {  this.rating = rating;  }  @Override  public String toString() {  System.out.println(super.toString());  return "Rating ID : " + id + "\nReview : " + review + "\nRating Stars : " + stars;  }  }  public class Student{    private int id;  private String name;  private String department;  private int courseId;  public Student(int id, String name, String department, int courseId) {  super();  this.id = id;  this.name = name;  this.department = department;  this.courseId = courseId;  }  @Override  public String toString() {  return "Student :\nId : " + id + "\nName : " + name + "\nDepartment : " + department + "\nCourse Id : " + courseId;  }  } | INHERITANCE - BOOKING TICKETS | import java.io.BufferedReader;  import java.io.IOException;  import java.io.InputStreamReader;  public class Main {    public static void main(String args[]) throws IOException {  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));    System.out.println("Enter the name of the Aircraft");  String aircraftName=br.readLine();  System.out.println("Enter the source");  String source=br.readLine();  System.out.println("Enter the destination");  String destination = br.readLine();  System.out.println("Enter the type of Flight\n1.Public Aircraft\n2.Private Aircraft");  int choice=Integer.parseInt(br.readLine());    if(choice==1)  {  System.out.println("Is the flight check in before two hours");  String ans = br.readLine();  Boolean b1;  if(ans.equals("yes"))  {  b1=true;  }  else  {  b1=false;  }  System.out.println("Enter the number of kgs allowed per person");  int noOfKgsallowed=Integer.parseInt(br.readLine());    System.out.println("Enter the additional fee charged for extra baggage per Kg");  float additionalFeeperkg=Float.parseFloat(br.readLine());      Aircraft b = new PublicAircraft(aircraftName,source,destination,b1,noOfKgsallowed,additionalFeeperkg);  System.out.println("Flight Details :");  System.out.println("Public Aircraft:");  b.displayDetails();    }    if(choice==2)  {  System.out.println("Is the flight check in before two hours");  Boolean ans = Boolean.parseBoolean(br.readLine());  System.out.println("Enter the name of the pilot chose");  String pilotPreference=br.readLine();  System.out.println("Enter the Purpose of your flight");  String purpose=br.readLine();      Aircraft b = new PrivateAircraft(aircraftName,source,destination,ans,pilotPreference,purpose);  System.out.println("Flight Details :");  System.out.println("Private Aircraft:");  b.displayDetails();    }    }  }  public class PublicAircraft extends Aircraft {  private boolean checkinbeforetwohours;  private int noOfKgsallowed;  private float additionalFeeperkg;  String result;    public boolean isCheckinbeforetwohours() {  return checkinbeforetwohours;  }    public void setCheckinbeforetwohours(boolean checkinbeforetwohours) {  this.checkinbeforetwohours = checkinbeforetwohours;  }  public int getNoOfKgsallowed() {  return noOfKgsallowed;  }  public void setNoOfKgsallowed(int noOfKgsallowed) {  this.noOfKgsallowed = noOfKgsallowed;  }  public float getAdditionalFeeperkg() {  return additionalFeeperkg;  }  public void setAdditionalFeeperkg(float additionalFeeperkg) {  this.additionalFeeperkg = additionalFeeperkg;  }  public PublicAircraft(String aircraftName, String source, String destination, boolean checkinbeforetwohours,  int noOfKgsallowed, float additionalFeeperkg) {  super(aircraftName, source, destination);  this.checkinbeforetwohours = checkinbeforetwohours;  this.noOfKgsallowed = noOfKgsallowed;  this.additionalFeeperkg = additionalFeeperkg;  result = checkinbeforetwohours ? "Yes" : "No";  }    public void displayDetails () {  super.displayDetails();  System.out.println("Flight check in before two hours : "+result+"\nNumber of kgs allowed per person : "+noOfKgsallowed+"\nAdditional fee charged for extra baggage per Kg : "+additionalFeeperkg);  }  }  public class Aircraft {  protected String aircraftName;  protected String source;  protected String destination;  public String getAircraftName() {  return aircraftName;  }  public void setAircraftName(String aircraftName) {  this.aircraftName = aircraftName;  }  public String getSource() {  return source;  }  public void setSource(String source) {  this.source = source;  }  public String getDestination() {  return destination;  }  public void setDestination(String destination) {  this.destination = destination;  }  public Aircraft(String aircraftName, String source, String destination) {  super();  this.aircraftName = aircraftName;  this.source = source;  this.destination = destination;  }    public void displayDetails () {  System.out.println("Aircraft Name : "+aircraftName+"\nSource : "+source+"\nDestination : "+destination);  }  }  public class PrivateAircraft extends Aircraft {  private boolean checkinbeforetwohours;  private String pilotPreference;  private String purpose;  String result;    public boolean isCheckinbeforetwohours() {  return checkinbeforetwohours;  }  public void setCheckinbeforetwohours(boolean checkinbeforetwohours) {  this.checkinbeforetwohours = checkinbeforetwohours;  }  public String getPilotPreference() {  return pilotPreference;  }  public void setPilotPreference(String pilotPreference) {  this.pilotPreference = pilotPreference;  }  public String getPurpose() {  return purpose;  }  public void setPurpose(String purpose) {  this.purpose = purpose;  }  public PrivateAircraft(String aircraftName, String source, String destination, boolean checkinbeforetwohours,  String pilotPreference, String purpose) {  super(aircraftName, source, destination);  this.checkinbeforetwohours = checkinbeforetwohours;  this.pilotPreference = pilotPreference;  this.purpose = purpose;  result = checkinbeforetwohours ? "Yes" : "No";  }    public void displayDetails () {  super.displayDetails();  System.out.println("Flight check in before two hours : "+result+"\nPilot chose : "+pilotPreference+"\nPurpose of the flight : "+purpose);  }  } |  |
|  | GST CALCULATION | import java.util.Scanner;  public class Main{  public static void main(String[] args){  Scanner sc = new Scanner(System.in);  System.out.println("Enter event name");  String name = sc.nextLine();    System.out.println("Enter the cost per day");  double costPerDay = sc.nextDouble();  System.out.println("Enter the number of days");  int noOfDays = sc.nextInt();  System.out.println("Enter the type of event\n1.Exhibition\n2.Stage Event");  int ty = sc.nextInt();  String type = Integer.toString(ty);  Event e;  switch(type){  case "1":  System.out.println("Enter the number of stalls");  int noOfStalls = sc.nextInt();  e = new Exhibition(name, type, costPerDay, noOfDays, noOfStalls);  e.toString();  break;  case "2":  System.out.println("Enter the number of seats");  int noOfSeats = sc.nextInt();  e = new StageEvent(name, type, costPerDay, noOfDays,noOfSeats);  e.toString();  break;    default:  System.out.println("Invalid input");  }  }  }  class StageEvent extends Event{  private static int gst = 15;  private int noOfSeats;    public StageEvent(String name, String type, double costPerDay, int noOfDays, int noOfSeats){  super(name, type, costPerDay, noOfDays);  this.noOfSeats = noOfSeats;  }    public double totalCost() {  double totalCost = (costPerDay \*noOfDays)\*(100 + StageEvent.gst)/100 ;  return totalCost;  }    public String toString() {  System.out.println("Event Details");  System.out.format("Name:%s%n",name);  System.out.println("Type:Stage Event");  System.out.format("Number of seats:%d%n",noOfSeats);  System.out.format("Total amount:%.2f%n",totalCost());  return "";  }  }  class Exhibition extends Event {  private static int gst = 5;  private int noOfStalls;    public Exhibition(String name, String type, double costPerDay, int noOfDays, int noOfStalls){  super(name, type, costPerDay, noOfDays);  this.noOfStalls = noOfStalls;  }    public double totalCost() {  double totalCost = (costPerDay \*noOfDays)\*(100 + Exhibition.gst)/100;  return totalCost;  }    public String toString() {  System.out.println("Event Details");  System.out.format("Name:%s%n",name);  System.out.println("Type:Exhibition");  System.out.format("Number of stalls:%d%n",noOfStalls);  System.out.format("Total amount:%.2f%n",totalCost());  return "";  }  }  class Event{  protected String name;  protected String type;  protected double costPerDay;  protected int noOfDays;    public Event(){  }    public Event(String name, String type, double costPerDay, int noOfDays){  this.name = name;  this.type = type;  this.costPerDay = costPerDay;  this.noOfDays = noOfDays;  }  } | OVERLOADING MAKEPAYMENT() | import java.util.\*;  public class Main {  public static void main(String[] args){  String stageEvent;  String customer;  String noOfSeats\_s;  int noOfSeats;  Scanner scan = new Scanner(System.in);  System.out.println("Enter the Booking details");  String csv = scan.nextLine();  String[] stringarray = csv.split(",");  stageEvent = stringarray[0];  customer = stringarray[1];  noOfSeats\_s = stringarray[2];  noOfSeats=Integer.parseInt(noOfSeats\_s);  System.out.println("Payment mode\n"+ "1.Cash payment\n"+ "2.Wallet payment\n"+ "3.Credit card payment");  int option = scan.nextInt();  TicketBooking tb = new TicketBooking(stageEvent,customer,noOfSeats);  double amount;  if(option==1) {  System.out.println("Enter the amount");  amount = scan.nextDouble();  System.out.println(tb.toString());  tb.makePayment(amount);  } else if(option==2) {  System.out.println("Enter the amount");  amount = scan.nextDouble();  System.out.println("Enter the wallet number");  scan.nextLine();  String walletNumber = scan.nextLine();  System.out.println(tb.toString());  tb.makePayment(walletNumber,amount);  }else if(option==3) {  System.out.println("Enter card holder name");  scan.nextLine();  String name= scan.nextLine();  System.out.println("Enter the amount");  amount = scan.nextDouble();  System.out.println("Enter the credit card type");  scan.nextLine();  String c\_type = scan.nextLine();  System.out.println("Enter the CCV number");  String ccv = scan.nextLine();  System.out.println(tb.toString());  tb.makePayment(c\_type,ccv,name,amount);  }else {  System.out.println("Invalid choice");  }  scan.close();  }  }  import java.util.Scanner;  public class TicketBooking {  private String stageEvent;  private String customer;  private int noOfSeats;  public TicketBooking(String stageEvent, String customer, int noOfSeats) {  this.stageEvent = stageEvent;  this.customer = customer;  this.noOfSeats = noOfSeats;  }  public String getStageEvent() {  return stageEvent;  }  public void setStageEvent(String stageEvent) {  this.stageEvent = stageEvent;  }  public String getCustomer() {  return customer;  }  public void setCustomer(String customer) {  this.customer = customer;  }  public int getNoOfSeats() {  return noOfSeats;  }  public void setNoOfSeats(int noOfSeats) {  this.noOfSeats = noOfSeats;  }  public void makePayment(Double amount) {  System.out.println("Amount " + amount +" paid in cash");  }  public void makePayment(String walletNumber ,Double amount) {  System.out.println("Amount " + amount +" paid using wallet number " + walletNumber);  }  @Override  public String toString() {  return "Stage event:" + stageEvent + "\nCustomer:" + customer + "\nNumber of seats:" + noOfSeats;  }  public void makePayment(String creditCard,String ccv,String name,Double amount) {  System.out.println("Holder name:" + name + "\nAmount "+ amount +" paid using " + creditCard +" card"  + "\nCCV:" +ccv);  }  } | ABSTRACT CLASS - INTRODUCTION | import java.io.BufferedReader;  import java.io.IOException;  import java.io.InputStreamReader;  public class Main {  public static void main(String[] args) throws NumberFormatException, IOException {  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));  System.out.println("List of Shapes:\n1.Circle\n2.Rectangle\n3.Square\nEnter your choice:");  int choice = Integer.parseInt(br.readLine());  switch(choice){  case 1:  System.out.println("Enter the radius of the Circle:");  Float r = Float.parseFloat(br.readLine());  Circle c = new Circle(r);  System.out.printf("The perimeter is %.2f",c.calculatePerimeter());  break;  case 2:  System.out.println("Enter the length of the Rectangle:");  Float l = Float.parseFloat(br.readLine());  System.out.println("Enter the breadth of the Rectangle:");  Float b = Float.parseFloat(br.readLine());  Rectangle rec = new Rectangle(l, b);  System.out.printf("The perimeter is %.2f",rec.calculatePerimeter());  break;  case 3:  System.out.println("Enter the side of the Square:");  Float s = Float.parseFloat(br.readLine());  Square sq = new Square(s);  System.out.printf("The perimeter is %.2f",sq.calculatePerimeter());  break;  default:  System.out.println("Enter a valid choice");  }  }  }  public abstract class Shape {  public abstract Double calculatePerimeter();  }  public class Square extends Shape {  private Float side;    public Float getSide() {  return side;  }  public void setSide(Float side) {  this.side = side;  }  public Square(Float side) {  super();  this.side = side;  }  @Override  public Double calculatePerimeter() {  return (double) (4\*side);  }    }  public class Circle extends Shape {  private final float pi = 3.14f;  private Float radius;    public Float getRadius() {  return radius;  }  public void setRadius(Float radius) {  this.radius = radius;  }    public Circle(Float radius) {  super();  this.radius = radius;  }  @Override  public Double calculatePerimeter() {  return (double)(2\*pi\*radius);  }    }  public class Rectangle extends Shape {  private Float length;  private Float bredth;  public Float getLength() {  return length;  }  public void setLength(Float length) {  this.length = length;  }  public Float getBredth() {  return bredth;  }  public void setBredth(Float bredth) {  this.bredth = bredth;  }  public Rectangle(Float length, Float bredth) {  super();  this.length = length;  this.bredth = bredth;  }  @Override  public Double calculatePerimeter() {  return (double)((length+bredth)\*2);  }  } |  |
|  | ABSTRACT EVENT | import java.util.Scanner;  public class Main {  public static void main(String[] args) throws Exception{  Scanner sc = new Scanner(System.in);  Event ev;  System.out.println("Enter your choice");  System.out.println("1.Exhibition");  System.out.println("2.StageEvent");  int ch = Integer.parseInt(sc.nextLine());  switch(ch){  case 1:  System.out.println("Enter the details in CSV format");  String resp1 = sc.nextLine();  String[] arr1 = resp1.split(",");  ev = new Exhibition(arr1[0], arr1[1], arr1[2], arr1[3], Integer.parseInt(arr1[4]), Double.parseDouble(arr1[5]));  ev.toString();  break;  case 2:  System.out.println("Enter the details in CSV format");  String resp2 = sc.nextLine();  String[] arr2 = resp2.split(",");  ev = new StageEvent(arr2[0], arr2[1], arr2[2], arr2[3], Integer.parseInt(arr2[4]) ,Double.parseDouble(arr2[5]));  ev.toString();  break;  default:  System.out.println("Invalid choice");  }  sc.close();  }  }  public abstract class Event {  protected String name;  protected String detail;  protected String type;  protected String organiser;  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public String getDetail() {  return detail;  }  public void setDetail(String detail) {  this.detail = detail;  }  public String getType() {  return type;  }  public void setType(String type) {  this.type = type;  }  public String getOrganiser() {  return organiser;  }  public void setOrganiser(String organiser) {  this.organiser = organiser;  }  public Event(String name, String detail, String type, String organiser){  this.name=name;  this.detail=detail;  this.type=type;  this.organiser=organiser;  }  public abstract double calculateAmount();  public String toString(){  return "";  }  }  public class StageEvent extends Event {  private Integer noOfShows;  private Double costPerShow;  public StageEvent(String name, String detail, String type,String organiser,Integer noOfShows,Double costPerShow){  super(name, detail, type, organiser);  this.noOfShows = noOfShows;  this.costPerShow = costPerShow;  }  public double calculateAmount() {  return noOfShows \* costPerShow;  }  public String toString(){  System.out.println("Stage Event Details");  System.out.format("Event Name:%s%n",name);  System.out.format("Detail:%s%n",detail);  System.out.format("Type:%s%n",type);  System.out.format("Organiser Name:%s%n",organiser);  System.out.format("Total Cost:%.1f%n",calculateAmount());  return "";  }  }  public class Exhibition extends Event {  private Integer noOfStalls;  private Double rentPerStall;  public Exhibition(String name, String detail, String type,String organiser,Integer noOfStalls, Double rentPerStall){  super(name, detail, type, organiser);  this.noOfStalls = noOfStalls;  this.rentPerStall = rentPerStall;  }  public double calculateAmount() {  return noOfStalls \* rentPerStall;  }  public String toString(){  System.out.println("Exhibition Details");  System.out.format("Event Name:%s%n",name);  System.out.format("Detail:%s%n",detail);  System.out.format("Type:%s%n",type);  System.out.format("Organiser Name:%s%n",organiser);  System.out.format("Total Cost:%.1f%n",calculateAmount());  return "";  }  } |  |  |  |  |  |
|  | OVERRIDING-SIMPLE | import java.io.IOException;  import java.util.Scanner;  public class Main {  public static void main(String[] args) throws IOException {  Scanner sc = new Scanner(System.in);  System.out.println("Enter the name of the event:");  String name = sc.nextLine();  System.out.println("Enter the detail of the event:");  String detail = sc.nextLine();  System.out.println("Enter the owner name of the event:");  String ownerName = sc.nextLine();  System.out.println("Enter the type of the event:\n1.Exhibition\n2.StageEvent");  String type = sc.next();  Event ev ;  switch(type){  case "1":  System.out.println("Enter the number of stalls:");  int noOfStalls = sc.nextInt();  ev = new Exhibition(name, detail, ownerName, noOfStalls);  System.out.format("The projected revenue of the event is %.1f%n",ev.projectedRevenue());  break;  case "2":  System.out.println("Enter the number of shows:");  int noOfShows = sc.nextInt();  System.out.println("Enter the number of seats per show:");  int noOfSeatsPerShow = sc.nextInt();  ev = new StageEvent(name, detail, ownerName, noOfShows, noOfSeatsPerShow);  System.out.format("The projected revenue of the event is %.1f%n",ev.projectedRevenue());  break;  default:  System.out.println("Invalid input");  }  }  }  public abstract class Event {  protected String name;  protected String detail;  protected String ownerName;  public Event(String name, String detail, String ownerName)  {  this.name = name;  this.detail = detail;  this.ownerName = ownerName;  }  public abstract Double projectedRevenue();  }  public class StageEvent extends Event{  private Integer noOfShows;  private Integer noOfSeatsPerShow;  public StageEvent(String name, String detail, String ownerName,Integer noOfShows,Integer noOfSeatsPerShow)  {  super(name,detail,ownerName);  this.noOfShows = noOfShows;  this.noOfSeatsPerShow = noOfSeatsPerShow;  }  public Double projectedRevenue()  {  return noOfShows\*noOfSeatsPerShow\*50.0;  }  }  public class Exhibition extends Event {  //Your code here  private Integer noOfStalls;  public Exhibition(String name, String detail, String ownerName,Integer noOfStalls)  {  super(name, detail, ownerName);  this.noOfStalls = noOfStalls;  }  public Double projectedRevenue()  {  return noOfStalls \*10000.0;  }  } |  |  |  |  |  |
| Interface and Inner Class | INTERFACES | import java.util.Scanner;  import java.text.DecimalFormat;  public class Main  {  public static void main(String args[])  {  System.out.println("1.Current Account");  System.out.println("2.Savings Account");    Scanner sc =new Scanner(System.in);  int type = sc.nextInt();    System.out.println("Name");  String name =sc.next();    System.out.println("Account Number");  String acNo =sc.next();    System.out.println("Account Balance");  double acBalance = sc.nextDouble();    System.out.println("Enter the Start Date(yyyy-mm-dd)");  String date =sc.next();    System.out.println("Enter the Years");  int n = sc.nextInt();    DecimalFormat deci = new DecimalFormat("0.00");    if (type == 1) {  CurrentAccount ca = new CurrentAccount();    System.out.println("Maintenance Charge For Current Account \n"+deci.format(ca.calculateMaintenanceCharge(n)));  }    if (type == 2) {  SavingsAccount sa = new SavingsAccount();    System.out.println("Maintenance Charge For Savings Account \n"+deci.format(sa.calculateMaintenanceCharge(n)));  }  }  }  public class CurrentAccount implements MaintainanceCharge  {  // @Override  public float calculateMaintenanceCharge(float noOfYears) {    float charge = (100\*noOfYears)+200;  return charge;  }  }  class SavingsAccount implements MaintainanceCharge  {  //@Override  public float calculateMaintenanceCharge(float noOfYears) {    float charge = (2\*50\*noOfYears)+50;  return charge;  }  }  class Account  {  private String name;  private String accountNumber;  private double balance;  private String startDate;    public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }    public String getAccountNumber() {  return accountNumber;  }  public void setAccountNumber(String accountNumber) {  this.accountNumber = accountNumber;  }    public double getBalance() {  return balance;  }  public void setBalance(double balance) {  this.balance = balance;  }    public String getStartDate() {  return startDate;  }  public void setStartDate(String startDate) {  this.startDate = startDate;  }    public Account(String name, String accountNumber, double balance, String startDate) {  super();  this.name = name;  this.accountNumber = accountNumber;  this.balance = balance;  this.startDate = startDate;  }      }  interface MaintainanceCharge{  float calculateMaintenanceCharge(float noOfYears);  } | MULTIPLE INHERITANCE WITH INTERFACES | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("Enter the distance travelled : ");  int distance = sc.nextInt();  System.out.println("Enter the speed of the vehicle : ");  int speed = sc.nextInt();  Bike b = new Bike(distance,speed);  System.out.println("Total distance travelled : "+b.totalDistance());  System.out.println("Average speed maintained : "+b.averageSpeed());  sc.close();  }  }  public interface BikeSpeed{  public int averageSpeed();  }  public class Bike implements BikeDistance, BikeSpeed {  private int distance;  private int speed;    public int getDistance() {  return distance;  }  public void setDistance(int distance) {  this.distance = distance;  }  public int getSpeed() {  return speed;  }  public void setSpeed(int speed) {  this.speed = speed;  }  public Bike() {  super();  }  public Bike(int distance, int speed) {  super();  this.distance = distance;  this.speed = speed;  }  @Override  public int averageSpeed() {  return distance;  }  @Override  public int totalDistance() {  return distance\*speed;  }  }  public interface BikeDistance{  public int totalDistance();  } | ROAD FIGHTER(INTERFACE) | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  int life = 5;  int Damagelevel = 0;    Villain v = new Villain(Damagelevel);  Hero h = new Hero(life);  Scanner sc = new Scanner(System.in);    while((life != 0)&&(Damagelevel !=100)){    System.out.println("Enter who punches :");  System.out.println("1.Hero");  System.out.println("2.Villain");    int person = sc.nextInt();    if(person== 2) {  life = life-1;  h.status();    }  else {  v.status();  Damagelevel = Damagelevel+50;  }  }    sc.close();  if(life == 0) {  System.out.println("You lose the GAME !!!");  }    else {  System.out.println("----GAME OVER----\nYOU WINS!!!");  }  }  }  interface GameStatus {  public void status();  }  import java.util.Scanner;  class Hero implements GameStatus{  private int life ;    public Hero(int life) {  super();  this.life = life;  }  @Override  public void status() {  life = life -1;  System.out.println("You have "+life+" lives left");    }  public int getLife() {  return life;  }  public void setLife(int life) {  this.life = life;  }    }  class Villain implements GameStatus{  private int damage;  public Villain(int damage) {  super();  this.damage = damage;  }    @Override  public void status() {  damage = damage+50;  System.out.println("The Villain Damage level is :"+damage);    }  public int getDamage() {  return damage;  }  public void setDamage(int damage) {  this.damage = damage;  }  } |  |
|  | INTERFACE | import java.util.Scanner;  import java.io.IOException;  public class Main {  public static void main(String[] args) throws IOException{    String Stallname;  Integer cost;  String owner;  Scanner sc = new Scanner(System.in);    System.out.println("Choose Stall Type\n1)Gold Stall\n2)Premium Stall\n3)Executive Stall");  String type = sc.next();    Stall s;  switch(type){  case "1":  System.out.println("Enter Stall details in comma separated(Stall Name,Stall Cost,Owner Name,Number of TV sets)");  sc.nextLine();  String resp1 = sc.nextLine();  String[] arr1=resp1.split(",");  Stallname = arr1[0];  cost = Integer.parseInt(arr1[1]);  owner = arr1[2];  Integer tvset = Integer.parseInt(arr1[3]);  s= new GoldStall(Stallname,cost,owner,tvset);  s.display();  break;    case "2":  System.out.println("Enter Stall details in comma separated(Stall Name,Stall Cost,Owner Name,Number of Projectors)");  sc.nextLine();  String resp2 = sc.nextLine();  String[] arr2 = resp2.split(",");  Stallname = arr2[0];  cost = Integer.parseInt(arr2[1]);  owner = arr2[2];  Integer projector = Integer.parseInt(arr2[3]);  s= new PremiumStall(Stallname,cost,owner,projector);  s.display();  break;    case "3":  System.out.println("Enter Stall details in comma separated(Stall Name,Stall Cost,Owner Name,Number of Screens)");  sc.nextLine();  String resp3 = sc.nextLine();  String[] arr3 = resp3.split(",");  Stallname = arr3[0];  cost = Integer.parseInt(arr3[1]);  owner = arr3[2];  Integer screen = Integer.parseInt(arr3[3]);  s= new ExecutiveStall(Stallname,cost,owner,screen);  s.display();  break;  default:  System.out.println("Invalid Stall Type");  }  }  }  public interface Stall {    public abstract void display();  }  public class PremiumStall implements Stall{  private String stallName;  private Integer cost;  private String ownerName;  private Integer projector;      public String getStallName() {  return stallName;  }  public void setStallName(String stallName) {  this.stallName = stallName;  }  public Integer getCost() {  return cost;  }  public void setCost(Integer cost) {  this.cost = cost;  }  public String getOwnerName() {  return ownerName;  }  public void setOwnerName(String ownerName) {  this.ownerName = ownerName;  }  public Integer getProjector() {  return projector;  }  public void setProjector(Integer projector) {  this.projector = projector;  }  public PremiumStall(){}  public PremiumStall(String stallName, Integer cost, String ownerName, Integer projector) {  super();  this.stallName = stallName;  this.cost = cost;  this.ownerName = ownerName;  this.projector = projector;  }  // @Override  public void display() {  System.out.println("Stall Name:"+stallName);  System.out.println("Cost:"+cost+".Rs");  System.out.println("Owner Name:"+ownerName);  System.out.println("Number of Projectors:"+projector);    }  }  public class ExecutiveStall implements Stall{  private String stallName;  private Integer cost;  private String ownerName;  private Integer screen;      public String getStallName() {  return stallName;  }  public void setStallName(String stallName) {  this.stallName = stallName;  }  public Integer getCost() {  return cost;  }  public void setCost(Integer cost) {  this.cost = cost;  }  public String getOwnerName() {  return ownerName;  }  public void setOwnerName(String ownerName) {  this.ownerName = ownerName;  }  public Integer getscreen() {  return screen;  }  public void setScreen(Integer screen) {  this.screen = screen;  }  public ExecutiveStall(){}  public ExecutiveStall(String stallName, Integer cost, String ownerName, Integer screen) {  super();  this.stallName = stallName;  this.cost = cost;  this.ownerName = ownerName;  this.screen = screen;  }  public String toString() {  return "stallDemo [stallName=" + stallName + ", cost=" + cost + ", ownerName=" + ownerName + ", screen="  + screen + "]";  }  // @Override  public void display() {  System.out.println("Stall Name:"+stallName);  System.out.println("Cost:"+cost+".Rs");  System.out.println("Owner Name:"+ownerName);  System.out.println("Number of Screens:"+screen);    }  }  public class GoldStall implements Stall{  private String stallName;  private Integer cost;  private String ownerName;  private Integer tvSet;      public String getStallName() {  return stallName;  }  public void setStallName(String stallName) {  this.stallName = stallName;  }  public Integer getCost() {  return cost;  }  public void setCost(Integer cost) {  this.cost = cost;  }  public String getOwnerName() {  return ownerName;  }  public void setOwnerName(String ownerName) {  this.ownerName = ownerName;  }  public Integer getTvSet() {  return tvSet;  }  public void setTvSet(Integer tvSet) {  this.tvSet = tvSet;  }  public GoldStall(){}  public GoldStall(String stallName, Integer cost, String ownerName, Integer tvSet) {  super();  this.stallName = stallName;  this.cost = cost;  this.ownerName = ownerName;  this.tvSet = tvSet;  }    @Override  public void display() {  System.out.println("Stall Name:"+stallName);  System.out.println("Cost:"+cost+".Rs");  System.out.println("Owner Name:"+ownerName);  System.out.println("Number of TV sets:"+tvSet);    }  } |  |  | NESTED CLASSES | import java.util.Scanner;  public class Main {  public static void main(String[] args) {    Scanner sc = new Scanner(System.in);  System.out.println("Enter Stall details in comma separated(Stall Name,Stall Description,Owner Name,Stall Cost,Number of TV set,Number of Projectors)");  String s1 = sc.nextLine();    String [] input = s1.split(",");    String name = input[0];  String details = input[1];  String owner = input[2];  int cost = Integer.parseInt(input[3]);  int tv = Integer.parseInt(input[4]);  int proj = Integer.parseInt(input[5]);    Stall c1 =new Stall(name, details, owner, cost);  Stall.GoldStall g = c1.new GoldStall(tv);  Stall.GoldStall.PlatinumStall p = g.new PlatinumStall(proj);  p.display();  sc.close();  }  }  public class Stall {  public String name;  public String detail;  public String owner;  public int cost;    public Stall(String name, String detail, String owner, int cost) {  super();  this.name = name;  this.detail = detail;  this.owner = owner;  this.cost = cost;  }    public class GoldStall {  public int tvSet;    public GoldStall(int tvSet) {  super();  this.tvSet = tvSet;    }    public class PlatinumStall {  public int projector;    public PlatinumStall(int projector) {  super();  this.projector = projector;  }  public void display(){    int total = cost+(projector\*500)+(tvSet\*100);  System.out.println("Stall Name:"+name);  System.out.println("Details:"+detail);  System.out.println("Owner Name:"+owner);  System.out.println("TV Sets:"+tvSet);  System.out.println("Projectors:"+projector);  System.out.println("Total Cost:" +total);  }  }  }  } |  |
|  | INTERFACE PRACTICAL PROBLEM 1 | import java.io.BufferedReader;  import java.io.IOException;  import java.io.InputStreamReader;  public class Main {  public static void main(String[] args) throws IOException {    System.out.println("Welcome to Notification Setup\nPlease select your bank:\n1)ICICI\n2)HDFC");  BufferedReader bf = new BufferedReader(new InputStreamReader(System.in));    int type = Integer.parseInt(bf.readLine());  BankFactory bank = new BankFactory();  switch(type) {    case 1:  System.out.println("Enter the type of Notification you want to enter\n1)SMS\n2)Mail\n3)Courier");  String op1 = bf.readLine();    switch(op1) {    case "1":  bank.getIcici().notificationBySms();  break;    case "2" :  bank.getIcici().notificationByEmail();  break;    case "3":  bank.getIcici().notificationByCourier();  break;  default:  System.out.println("Invalid Input");  }  break;    case 2:  System.out.println("Enter the type of Notification you want to enter\n1)SMS\n2)Mail\n3)Courier");  String op2 = bf.readLine();    switch(op2) {    case "1":  bank.getHdfc().notificationBySms();  break;    case "2" :  bank.getHdfc().notificationByEmail();  break;    case "3":  bank.getHdfc().notificationByCourier();  break;    default:  System.out.println("Invalid Input");  }  break;  default:  System.out.println("Invalid Input");  }  }  }  public class HDFC implements Notification {  @Override  public void notificationBySms() {  System.out.println("HDFC - Notification By SMS");  }  @Override  public void notificationByEmail() {  System.out.println("HDFC - Notification By Mail");  }  @Override  public void notificationByCourier() {  System.out.println("HDFC - Notification By Courier");  }  }  public class BankFactory {    public ICICI getIcici() {  ICICI icici = new ICICI();  return icici;  }    public HDFC getHdfc() {  HDFC hdfc = new HDFC();  return hdfc;  }  }  public class ICICI implements Notification {  @Override  public void notificationBySms() {  System.out.println("ICICI - Notification By SMS");  }  @Override  public void notificationByEmail() {  System.out.println("ICICI - Notification By Mail");  }  @Override  public void notificationByCourier() {  System.out.println("ICICI - Notification By Courier");  }  }  public interface Notification{  public void notificationBySms();  public void notificationByEmail();  public void notificationByCourier();  } |  |  |  |  |  |
|  | STATIC INNER CLASS | import java.io.IOException;  import java.text.DecimalFormat;  import java.util.Scanner;  public class Main {  public static void main(String[] args) throws IOException {  Scanner sc = new Scanner(System.in);  System.out.println("Enter the shape\r\n1.Rectangle\r\n2.Triangle");  int ch = sc.nextInt();  DecimalFormat df = new DecimalFormat("0.00");  switch (ch) {  case 1:  System.out.println("Enter the length and breadth:");  Shape.value1 = sc.nextDouble();  Shape.value2 = sc.nextDouble();  Shape.Rectangle rec = new Shape.Rectangle();  Double rArea = rec.computeRectangleArea();  System.out.println("Area of rectangle is " + df.format(rArea));  break;  case 2:  System.out.println("Enter the base and height:");  Shape.value1 = sc.nextDouble();  Shape.value2 = sc.nextDouble();  Shape.Triangle tri = new Shape.Triangle();  Double tArea = tri.computeTriangleArea();  System.out.println("Area of triangle is " + df.format(tArea));  break;  default:  System.out.println("Invalid choice");  break;  }  }  }  public class Shape {    public static Double value1;  public static Double value2;  public static class Rectangle {  public Double computeRectangleArea() {  return (value1 \* value2);  }  }    public static class Triangle {  public Double computeTriangleArea() {  return ((0.5) \* (value1 \* value2));  }  }  } |  |  |  |  |  |
| Exception Handling | EXCEPTION INTRODUCTION | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int n=0;  System.out.println("Enter an integer input");  try {  n = sc.nextInt();  System.out.print("Entered value is "+n);  }  catch(Exception e) {  System.out.println(e);  }  sc.close();  }  } | EVENTTYPEDOESNOTEXISTSEXCEPTION |  | INPUT MISMATCH EXCEPTION | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int in=0;  System.out.println("Enter an integer input");  try {  in = sc.nextInt();  System.out.println("Entered value is "+in);  } catch (Exception e) {  System.out.println(e);  }  sc.close();  }  } |  |
|  | ARITHMETIC EXCEPTION | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  int cost,n;  System.out.println("Enter the cost of the item for n days");  cost = sc.nextInt();  System.out.println("Enter the value of n");  n = sc.nextInt();  try {  int pd = cost/n;  System.out.print("Cost per day of the item is "+pd);  }  catch(ArithmeticException e) {  System.out.println(e);  }  catch(Exception e) {  System.out.println(e);  }    sc.close();  }  } |  |  | WEAK PASSWORD EXCEPTION | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("Enter the user details");  String details = sc.nextLine();  String [] records = details.split(",");  User u = new User(records[0], records[1], records[2], records[3]);  try {  UserBO.validate(u);  System.out.println("Name:"+u.getName()+"\nMobile:"+u.getMobile()+"\nUsername:"+u.getUsername()+"\nPassword:"+u.getPassword());  } catch (WeakPasswordException e) {  System.out.println(e);  }    }  }  public class WeakPasswordException extends Exception {  public WeakPasswordException(String msg) {  super(msg);  }    }  public class UserBO {  static void validate(User u) throws WeakPasswordException {  boolean len=true,digit=false,charac=true,let=false;  String pass = u.getPassword();  if(pass.length()>20||pass.length()<10)  len=false;  if(pass.matches(".\*\\d.\*"))  digit=true;  if(pass.matches("[A-Za-z0-9]+"))  charac=false;  if(pass.matches(".\*[A-Za-z].\*"))  let = true;    if(len==false||digit==false||charac==false||let==false)  throw new WeakPasswordException("Your password is weak");  }  }  public class User {  private String name;  private String mobile;  private String username;  private String password;  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public String getMobile() {  return mobile;  }  public void setMobile(String mobile) {  this.mobile = mobile;  }  public String getUsername() {  return username;  }  public void setUsername(String username) {  this.username = username;  }  public String getPassword() {  return password;  }  public void setPassword(String password) {  this.password = password;  }  public User() {  super();  }  public User(String name, String mobile, String username, String password) {  super();  this.name = name;  this.mobile = mobile;  this.username = username;  this.password = password;  }  @Override  public String toString() {  return "Name:" + name + "\nMobile:" + mobile + "\nUsername:" + username + "\nPassword:" + password;  }  } |  |
|  | ARRAYINDEXOUTOFBOUNDSEXCEPTION | import java.util.Scanner;  public class Main {  public static void main(String args[]) {  Scanner sc = new Scanner(System.in);  System.out.println("Enter the number of seats to be booked:");  int size, num, count=1;  size = sc.nextInt();  int arr[] = new int[100];  try{  while(count<=size) {  System.out.println("Enter the seat number " +count);  num = sc.nextInt();  arr[num-1] = num;  count++;  }    System.out.println("The seats booked are:");  for(int i=0; i<100; i++) {  if(arr[i]!=0)  System.out.println(arr[i]);  }  }  catch(ArrayIndexOutOfBoundsException e) {  System.out.println(e);  }  sc.close();  }  } |  |  |  |  |  |
|  | PARSE EXCEPTION | import java.text.ParseException;  import java.text.SimpleDateFormat;  import java.util.Date;  import java.util.Scanner;  public class Main {  public static void main(String[] args) {    Scanner sc = new Scanner(System.in);  SimpleDateFormat sd = new SimpleDateFormat("dd-MM-yyyy-HH:mm:ss");  System.out.println("Enter the stage event start date and end date");    try {  String date1 = sc.next();  Date date3=sd.parse(date1);    String date2 = sc.next();  Date date4=sd.parse(date2);    Date d1=new Date();  d1=date3;    Date d2=new Date();  d2=date4;    System.out.println("Start date:" + sd.format(d1));  System.out.println("End date:" + sd.format(d2));  }  catch(ParseException e) {  System.out.println("Input dates should be in the format 'dd-MM-yyyy-HH:mm:ss'");  }  sc.close();  }  } |  |  |  |  |  |
|  | SEATNOTAVAILABLEEXCEPTION | import java.util.Scanner;  import java.io.\*;  public class Main {  public static void main(String args[]) throws SeatNotAvailableException{    Scanner sc = new Scanner(System.in);  System.out.println("Enter the number of rows and columns of the show:");  int n = sc.nextInt();  int size = (n\*n);  int s;  int a[] = new int[size];  int mat[][] = new int[n][n];    System.out.println("Enter the number of seats to be booked:");  int seats = sc.nextInt();  try {  for(int i=0;i<seats;i++) {  System.out.println("Enter the seat number "+(i+1));  s = sc.nextInt();    if(a[s]==0) {  a[s] =1;  int t=0;  for(int j=0;j<n;j++) {  for(int k=0;k<n;k++) {  mat[j][k] = a[t];  t++;  }  }  }else {  throw new SeatNotAvailableException("Already Booked");  }    }    }catch (Exception e) {  System.out.println(e);  }  finally {  System.out.println("The seats booked are:");  for(int i=0;i<n;i++) {  for(int j=0;j<n;j++) {  System.out.print(mat[i][j]+" ");  }  System.out.println();  }  }  }  }  public class SeatNotAvailableException extends Exception {    public SeatNotAvailableException(String msg) {  super(msg);  }  } |  |  |  |  |  |
| Java Memory Management, Threads & Reflection | ARTICLE COUNT | import java.io.\*;  public class Main {  public static void main(String[] args) throws Exception{  int count = 0;  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));  System.out.println("Enter the number of lines");  int lines = Integer.parseInt(br.readLine());  for(int i=1;i<=lines;i++)  {  System.out.println("Enter line "+i);  String line = br.readLine();  Article obj = new Article(line);  obj.start();  obj.join();  count += obj.getCount();    }  System.out.printf("There are %d articles in the given input\n",count);    }  }  public class Article extends Thread{  private String line;  private int count;    Article(String line)  {  this.line = line;  }  public String getLine() {  return line;  }  public void setLine(String line) {  this.line = line;  }  public Integer getCount() {  return count;  }  public void setCount(Integer count) {  this.count = count;  }  public void run()  {  int articleCount = 0;  String[] str = getLine().split(" ");  for(String s : str) {  if(s.equalsIgnoreCase("the")){  articleCount++;  }  if(s.equalsIgnoreCase("a")){  articleCount++;  }  if(s.equalsIgnoreCase("an")){  articleCount++;  }  }  setCount(articleCount);  }  } | CITY COUNT |  | USER NOTIFICATION |  |  |
|  | PROFIT OR LOSS | import java.io.BufferedReader;  import java.io.IOException;  import java.io.InputStreamReader;  import java.util.ArrayList;  import java.util.List;    public class Main {  public static void main(String args[]) throws NumberFormatException, IOException {    System.out.println("Enter the number of events");  BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  int noOfEvents = Integer.parseInt(in.readLine());    if(noOfEvents < 0 || noOfEvents % 2 != 0) {  System.out.println("Invalid Input");  System.exit(0);  }    ThreadGroup group = new ThreadGroup("Events-Group");    System.out.println("Enter event details in CSV");  int halfPt = noOfEvents / 2;      List <Event> events = new ArrayList<>();  for(int i = 0; i < halfPt; ++i) {  String [] eventDetails = in.readLine().split(",");  String eventName = eventDetails[0];  String hallName = eventDetails[1];  Double cost = Double.parseDouble(eventDetails[2]);  Integer hallCapacity = Integer.parseInt(eventDetails[3]);  Integer seatsBooked = Integer.parseInt(eventDetails[4]);    HallBooking booking = new HallBooking(hallName, cost, hallCapacity, seatsBooked);  Event event = new Event(eventName, booking);  events.add(event);  }  Thread t1 = new Thread(group, new ComputeStatus(events));  t1.start();  try {  t1.join();  } catch (InterruptedException e) {  e.printStackTrace();  }    events = new ArrayList<>();  for(int i = halfPt; i < noOfEvents; ++i) {  String [] eventDetails = in.readLine().split(",");  String eventName = eventDetails[0];  String hallName = eventDetails[1];  Double cost = Double.parseDouble(eventDetails[2]);  Integer hallCapacity = Integer.parseInt(eventDetails[3]);  Integer seatsBooked = Integer.parseInt(eventDetails[4]);    HallBooking booking = new HallBooking(hallName, cost, hallCapacity, seatsBooked);  Event event = new Event(eventName, booking);  events.add(event);  }  Thread t2 = new Thread(group, new ComputeStatus(events));  t2.start();  try {  t2.join();  } catch (InterruptedException e) {  e.printStackTrace();  }  }  }  public class Event {  private String name;  private HallBooking hallbooking;    public Event(String name, HallBooking hallbooking) {  super();  this.name = name;  this.hallbooking = hallbooking;  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public HallBooking getHallbooking() {  return hallbooking;  }  public void setHallbooking(HallBooking hallbooking) {  this.hallbooking = hallbooking;  }  }  public class HallBooking {  private String hallName;  private double cost;  private int hallCapacity;  private int seatsBooked;    public HallBooking(String hallName, double cost, int hallCapacity, int seatsBooked) {  super();  this.hallName = hallName;  this.cost = cost;  this.hallCapacity = hallCapacity;  this.seatsBooked = seatsBooked;  }  public String getHallName() {  return hallName;  }  public void setHallName(String hallName) {  this.hallName = hallName;  }  public double getCost() {  return cost;  }  public void setCost(double cost) {  this.cost = cost;  }  public int getHallCapacity() {  return hallCapacity;  }  public void setHallCapacity(int hallCapacity) {  this.hallCapacity = hallCapacity;  }  public int getSeatsBooked() {  return seatsBooked;  }  public void setSeatsBooked(int seatsBooked) {  this.seatsBooked = seatsBooked;  }  }  import java.util.List;  public class ComputeStatus implements Runnable{  private List<Event>eventList;    public ComputeStatus(List<Event> eventList) {  super();  this.eventList = eventList;  }    public List<Event> getEventList() {  return eventList;  }  public void setEventList(List<Event> eventList) {  this.eventList = eventList;  }  @Override  public void run() {  for(Event event : eventList) {  if(isProfitable(event)) {  System.out.println(event.getName() + " yields profit");  } else {  System.out.println(event.getName() + " yields loss");  }  }  }    private boolean isProfitable(Event event) {  HallBooking booking = event.getHallbooking();    Double value = ( 100.0 \* booking.getSeatsBooked())/ booking.getHallCapacity();  return value >= 60;  }  } |  |  | STALL REVENUE |  |  |
| Primitive Wrappers and java.util & java.lang classes | WRAPPER CLASS – INTEGER I | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("Enter an integer");  Integer num = sc.nextInt();  System.out.println("The binary equivalent of "+num+" is "+num.toBinaryString(num));  System.out.println("The hexadecimal equivalent of "+num+" is "+num.toHexString(num));  System.out.println("The octal equivalent of "+num+" is "+num.toOctalString(num));  System.out.println("Byte value of "+num+" is "+num.byteValue());  System.out.println("Short value of "+num+" is "+num.shortValue());  System.out.println("Long value of "+num+" is "+num.longValue());  System.out.println("Float value of "+num+" is "+num.floatValue());  System.out.println("Double value of "+num+" is "+num.doubleValue());  }  } | STRING API : ENDSWITH() : ILLUSTRATION |  | EMAIL ID COMPARISON |  |  |
|  | WRAPPER CLASS – 1 | public class Main {  public static void main(String[] args) {  System.out.println("Maximum exponent :"+Float.MAX\_EXPONENT+" ");  System.out.println("Maximum value :"+Float.MAX\_VALUE+" ");  System.out.println("Number of bits :"+Float.SIZE);  }  } | WRAPPER CLASS – INTEGER II |  | DATE - 2 |  |  |
|  | CONVERTING A STRING TO DOUBLE |  | STRING API : REPLACE() : ILLUSTRATION |  |  |  |  |
|  | STRING REVERSAL | import java.util.Scanner;  import java.io.\*;  public class Main {  public static void main(String[] args) throws IOException {  System.out.println("Enter a string to reverse");  Scanner sc = new Scanner(System.in);  String value = sc.nextLine();  StringBuilder revalue = new StringBuilder();  revalue.append(value);  System.out.println("Reverse of entered string is : "+revalue.reverse());  }  } |  |  |  |  |  |
|  | STRING API : STARTSWITH() : ILLUSTRATION | import java.util.Scanner;  import java.io.\*;  public class Main {  public static void main(String[] args) throws IOException {    System.out.println("Enter the string");  Scanner sc = new Scanner(System.in);  String value1 = sc.nextLine();    System.out.println("Enter the start string");  String value2 = sc.nextLine();    if(value1.startsWith(value2)){  System.out.println("\""+value1+"\" starts with \""+value2+"\"");  }  else  {  System.out.println("\""+value1+"\" does not start with \""+value2+"\"");  }  }  } |  |  |  |  |  |
|  | STRING API : SPLIT() : ILLUSTRATION | import java.util.StringTokenizer;  import java.util.Scanner;  import java.io.\*;  public class Main {  public static void main(String[] args) throws IOException {    Scanner sc = new Scanner(System.in);    System.out.println("Enter the string");    String value3 = sc.nextLine();    StringTokenizer st = new StringTokenizer(value3," ");  StringBuffer sb = new StringBuffer();    while(st.hasMoreElements()){    sb.append(st.nextElement()).append(" ");  }    String value4 = sb.toString();  System.out.println("The words in the string are");  String [] arr = value4.split(" ");    for(int i=0;i<arr.length;i++){    System.out.println(arr[i]);    }  }  } |  |  |  |  |  |
|  | STRING TOKENIZER | import java.util.Scanner;  import java.util.StringTokenizer;  import java.io.\*;  public class Main {  public static void main(String[] args) throws IOException {    Scanner sc = new Scanner(System.in);  String value = sc.nextLine();  String value2 = value.replaceAll("=", " ");  StringTokenizer st = new StringTokenizer(value2,";");    while(st.hasMoreTokens()){    System.out.println(st.nextToken());  }  }  } |  |  |  |  |  |
|  | CUSTOMER ADDRESS USING STRING BUILDER | import java.util.Scanner;  import java.io.\*;  public class Main {  public static void main(String[] args) throws IOException {  System.out.println("Enter Address Details :");  Scanner sc = new Scanner(System.in);    System.out.println("Enter Line 1 :");  String L1 = sc.nextLine();    System.out.println("Enter Line 2 :");  String L2 = sc.nextLine();    System.out.println("Enter City :");  String city = sc.nextLine();    System.out.println("Enter Country :");  String country = sc.nextLine();    System.out.println("Enter Zip Code :");  int zip = sc.nextInt();    Address address = new Address(L1, L2, city, country, zip);    address.toString();  }  }  public class Address {  private String line1;  private String line2;  private String city;  private String country;  private int zipCode;    public Address(String line1, String line2, String city, String country, int zipCode) {  super();  this.line1 = line1;  this.line2 = line2;  this.city = city;  this.country = country;  this.zipCode = zipCode;  }  public Address() {  super();  // TODO Auto-generated constructor stub  }  @Override  public String toString() {  StringBuilder sb = new StringBuilder();  StringBuilder add =sb.append("Address Details :\n" + line1 +",\n"+ line2 + ",\n" + city + " - "+zipCode+"\n" + country) ;  System.out.println(add);  return null;  }  public String getLine1() {  return line1;  }  public void setLine1(String line1) {  this.line1 = line1;  }  public String getLine2() {  return line2;  }  public void setLine2(String line2) {  this.line2 = line2;  }  public String getCity() {  return city;  }  public void setCity(String city) {  this.city = city;  }  public String getCountry() {  return country;  }  public void setCountry(String country) {  this.country = country;  }  public int getZipCode() {  return zipCode;  }  public void setZipCode(int zipCode) {  this.zipCode = zipCode;  }  } |  |  |  |  |  |
| Collections and Generics | ARRAYLIST - INTRODUCTION | import java.util.ArrayList;  import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  ArrayList<String> names = new ArrayList<>();  boolean result = false;  int i = 1;  do{  System.out.println("Enter the username " + i);  String name = sc.next();  names.add(name);  i++;  System.out.println("Do you want to continue?(y/n)");  String diss = sc.next();  result=diss.equals("y");  }while(result);  System.out.println("The Names entered are:");  for(String s:names)  {  System.out.println(s);  }  sc.close();  }  } | ITERATOR CLASS |  | LIST OF LIST |  |  |
|  | SET INTRODUCTION | import java.awt.List;  import java.util.LinkedHashSet;  import java.util.Scanner;  public class Main {  public static void main(String[] args){    boolean result=false;  LinkedHashSet<String> list = new LinkedHashSet<>();  int i=1;  Scanner scanner = new Scanner(System.in);    do{  System.out.println("Enter the username");  String name = scanner.nextLine();  list.add(name);  System.out.println("Do you want to continue?(Y/N)");  String diss = scanner.nextLine();  result=diss.equals("Y");  }while(result);    System.out.println("The unique number of usernames is "+list.size());    }  } | SORT() A LIST OF OBJECTS |  | MIN AND MAX |  |  |
|  | REVERSE() METHOD | import java.util.ArrayList;  import java.util.Collections;  import java.util.Formatter;  import java.util.Scanner;  import java.util.Iterator;  import java.util.\*;  public class Main {  public static void main(String args[]) {  ArrayList<User> au = new ArrayList<>();  Scanner scan = new Scanner(System.in);  Formatter f = new Formatter();  try{  System.out.println("Enter the number of users:");  int n = scan.nextInt();  scan.nextLine();    for(int i=0;i<n;i++) {  System.out.println("Enter the details of User "+ (i+1));  String input = scan.nextLine();  String[] stringarray = input.split(",");  String name = stringarray[0];  String mobileNumber = stringarray[1];  String username = stringarray[2];  String password =stringarray[3];  au.add(new User(name,mobileNumber,username,password));  }  Collections.sort(au);  Collections.reverse(au);  }catch(Exception e){    }  System.out.println("The user details in reverse order:");    System.out.printf("%-15s %-15s\n","Name ","Mobile number");  for(User a : au) {  System.out.printf("%-15s %-15s\n",a.getName(),a.getMobileNumber());  }  }  }  public class User implements Comparable{  //write your code here  private String name;  private String mobileNumber;  private String username;  private String password;  public User(String name, String mobileNumber, String username, String password) {  super();  this.name = name;  this.mobileNumber = mobileNumber;  this.username = username;  this.password = password;  }    public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public String getMobileNumber() {  return mobileNumber;  }  public void setMobileNumber(String mobileNumber) {  this.mobileNumber = mobileNumber;  }  public String getUsername() {  return username;  }  public void setUsername(String username) {  this.username = username;  }  public String getPassword() {  return password;  }  public void setPassword(String password) {  this.password = password;  }  @Override  public String toString() {  return name + mobileNumber ;  }  @Override  public int compareTo(Object o) {  String name1 = this.name;  String name2 = ((User)o).name;    return name1.compareTo(name2);  }  } |  |  | REPLICA OF A LIST |  |  |
|  | GENERIC CLASSES | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("Enter a integer :");  int val = sc.nextInt();  Item i = new Item(val);    System.out.println("Enter a string :");  sc.nextLine();  String st = sc.nextLine();  Item s = new Item(st);    System.out.println("Integer Value :"+i.getT()+"\nString Value :"+s.getT());  sc.close();  }  }  public class Item <T>{  T t;  Item(T t){  this.t = t;  }  public T getT() {  return t;  }  public void setT(T t) {  this.t = t;  }  } |  |  | STATE MAP |  |  |
|  |  |  |  |  | GENERIC METHODS |  |  |
| Java Streams and Writers | FILE HANDLING INTRODUCTION |  | READ AND WRITE |  | ITEM DETAILS |  |  |
|  | FILE WRITING | import java.io.IOException;  import java.util.List;  import java.util.Scanner;  import java.io.BufferedWriter;  import java.io.File;  import java.io.FileWriter;  import java.util.ArrayList;  import java.io.\*;  public class Main {  public static void main(String args[]) {  try{  Scanner sc = new Scanner(System.in);  ArrayList<User> users= new ArrayList<>();  System.out.println("Enter the number of users:");  int n = sc.nextInt();  sc.nextLine();  for(int i=0;i<n;i++) {  System.out.println("Enter the details of user : "+(i+1));  String input = sc.nextLine();  String[] stringarray = input.split(",");  String name = stringarray[0];  String mobileNo = stringarray[1];  String username =stringarray[3];  String password=stringarray[3];  users.add(new User(name,mobileNo,username,password));    }    File file = new File("output.csv");  if (!file.exists()) {  file.createNewFile();  }      FileWriter fw = new FileWriter(file);  BufferedWriter bw = new BufferedWriter(fw);  UserBO.writeFile(users, bw) ;      } catch (IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }catch (Exception e) {  // TODO Auto-generated catch block  e.printStackTrace();  }    }  }  import java.io.BufferedWriter;  import java.io.IOException;  import java.util.ArrayList;  import java.util.List;  public class UserBO {  public static void writeFile(ArrayList<User> userList, BufferedWriter bw) throws Exception {  try {      for(User t:userList) // unboxing  {  bw.write(t.getName()+","+t.getMobileNumber()+","+t.getUsername()+","+t.getPassword()+"\n");  }    bw.flush();  bw.close();      } catch (IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  }  }  public class User {    private String name;  private String mobileNumber;  private String username;  private String password;  public User(String name, String mobileNumber, String username, String password) {  super();  this.name = name;  this.mobileNumber = mobileNumber;  this.username = username;  this.password = password;  }  public User() {  super();  // TODO Auto-generated constructor stub  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public String getMobileNumber() {  return mobileNumber;  }  public void setMobileNumber(String mobileNumber) {  this.mobileNumber= mobileNumber;  }  public String getUsername() {  return username;  }  public void setUsername(String username) {  this.username = username;  }  public String getPassword() {  return password;  }  public void setPassword(String password) {  this.password = password;  }  } | ITEM COUNT |  | IO - SIMPLE FILE WRITE |  |  |
|  | HALL DETAILS | import java.util.ArrayList;  import java.util.List;  import java.util.Scanner;  public class Main {  public static void main(String[] args){    Scanner sc = new Scanner(System.in);  List<Hall> hall = new ArrayList<>();  System.out.println("Enter the number of halls:");  int n = sc.nextInt();  sc.nextLine();  for(int i=0;i<n;i++) {    String input = sc.nextLine();  String[] stringarray = input.split(",");  String name = stringarray[0];  String contact = stringarray[1];  double cost = Double.parseDouble(stringarray[2]);  String owner =stringarray[3];  hall.add(new Hall(name,contact,cost,owner));    }    Hall.writeHallDetails(hall);  }    }  import java.io.BufferedWriter;  import java.io.File;  import java.io.FileWriter;  import java.io.IOException;  import java.util.List;  public class Hall {  private String name;  private String contact;  private Double costPerDay;  private String owner;    static void writeHallDetails(List<Hall> halls) {  try {  File file = new File("hall.csv");  if (!file.exists()) {  file.createNewFile();  }      FileWriter fw = new FileWriter(file);  BufferedWriter bw = new BufferedWriter(fw);  for(Hall t:halls) // unboxing  {  bw.write(t.getName()+","+t.getContact()+","+t.getCostPerDay()+","+t.getOwner()+"\n");  }    bw.flush();  bw.close();      } catch (IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  }    public Hall(String name, String contact, Double costPerDay, String owner) {  super();  this.name = name;  this.contact = contact;  this.costPerDay = costPerDay;  this.owner = owner;  }  public Hall() {  super();  // TODO Auto-generated constructor stub  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public String getContact() {  return contact;  }  public void setContact(String contact) {  this.contact = contact;  }  public Double getCostPerDay() {  return costPerDay;  }  public void setCostPerDay(Double costPerDay) {  this.costPerDay = costPerDay;  }  public String getOwner() {  return owner;  }  public void setOwner(String owner) {  this.owner = owner;  }      } |  |  |  |  |  |
|  |  |  |  |  |  |  |  |