String

Q. to count the special characters

```
import java.util.Scanner;

public class Punctuation {

public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner sc= new Scanner(System.in);
        System.out.print("enter the string: ");
        String str=sc.nextLine();
        Punctuation strOne=new Punctuation();
        strOne.countPunctuation(str);

}
public int  countPunctuation(String str) {
int count=0;
for(int i=0;i<str.length();i++) {
char ch=str.charAt(i);
if(ch=='?'||ch=='.'||ch=='!'||ch==','||ch==';') {
                count=count+1;
        }
}
System.out.println("The number of special characters: "+count);
return count;
}

}
```

=-

Q.To replace vowels in the given string by character “b” using StringBuilder

```
import java.util.Scanner;

public class StringBuilderDemo {

        public String replace(String str,char ch[]) {

                for (int i = 0; i < str.length(); i++)
              {
                  if (ch[i]=='a'||ch[i]=='e'||ch[i]=='i'||ch[i]=='o'||ch[i]=='u')
                  {
                   ch[i]='b';
                  }
```

```
                }
                for (int i = 0; i < ch.length; i++) {

                System.out.print(ch[i]);
                }

                return "completed";
        }
        public static void main(String[] args) {
                // TODO Auto-generated method stub

                StringBuilder sb = new StringBuilder("");

        Scanner sc = new Scanner(System.in);

        System.out.print("enter string: ");
          String inputdata = sc.nextLine();
          sb.append(inputdata);
          String str = sb.toString();
          char[] ch=str.toCharArray();
          StringBuilderDemo a= new StringBuilderDemo();
          a.replace(str, ch);


        sc.close();

            }

    }
```

=-

3) package com.assignment.week2;

```
import java.util.Scanner;
public class StringTest {
    public static String concat(String str1,String str2) {
        String value=str1.concat(str2);
        System.out.println(value);
        return value;
    }
    public static int getIndex(String str,char ch) {
        int index= str.indexOf(ch);
        System.out.println(index);
        return ch;

    }
    public static String stringPadRight(String str,int len) {
        String result=String .format("%" + (-len) + "s", str).replace(" ", ",");
        System.out.println(result);
        return result;
```

```
	}
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner sc =new Scanner (System.in);
		StringTest a=new StringTest();

		System.out.println("string1");
		String str1=sc.next();
		System.out.println("string2");
		String str2=sc.next();
		a.concat(str1, str2);
		System.out.println("string");
		String str=sc.next();
		char c=str.charAt(0);
		System.out.println("input the element whose index is to be found");
		char ch= sc.next().charAt(0);
		System.out.println("enter length");
		int len=sc.nextInt();
		a.getIndex(str, ch);
		a.stringPadRight(str, len);
		sc.close();
}
}
=-=-=-=-=-=-=-=-=-=-=-=-=-=-==-=-=-=-=-=-=-=-=-=--=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
```

Q.to reverse the given input

```
 import java.util.Scanner;
public class StringMirror {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner sc=new Scanner(System.in);
		String input=sc.nextLine();
StringMirror a=new StringMirror();
a.getImage(input);
	}

	private void getImage(String input) {
		// TODO Auto-generated method stub
		StringBuffer sb=new StringBuffer();
		sb.append(input);
		System.out.println(input +"|" +sb.reverse());
	}
}
```

Q. to replace consonants in the given string

```
package javaProjectExamples;
```

```
import java.util.*;

public class ConsonatCount {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner sc=new Scanner(System.in);
		System.out.print("Enter a string: ");
		String str=sc.next();


		ConsonatCount ConsCount= new ConsonatCount();
	   System.out.println(ConsCount.StringReplace(str));

	}

	int count=0;
	public int StringReplace(String str) {
		char[] ch=str.toCharArray();

		for(int i=0;i<str.length();i++) {
			if(ch[i]=='b'||ch[i]=='c'||ch[i]=='d'||ch[i]=='f'||ch[i]=='g'||

ch[i]=='h'||ch[i]=='j'||ch[i]=='k'||ch[i]=='l'||ch[i]=='m'||ch[i]=='n'||

ch[i]=='p'||ch[i]=='q'||ch[i]=='r'||ch[i]=='s'||ch[i]=='t'||ch[i]=='v'||ch[i]=='w'||
					ch[i]=='x'||ch[i]=='y'||ch[i]=='z') {
				count=count+1;
			}
				}
		return count;
	}
}
```

=-

String Tokenizer

```
public class StringTokenizer {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		String s1="256@10,312@9,512@6";
		StringTokenizer st=new StringTokenizer(s1,",");
      //256@10
       // 312@9
       //512@6
	   while(st.hasMoreElements()) {
		String currentElement=(String )st.nextElement();

	}gst
```

```
	int sum=0;
		StringTokenizer st2=new StringTokenizer(s1,"@");

		while(st2.hasMoreElements()) {
			int price=Integer.parseInt((String)st2.nextElement());
			int quantity =Integer.parseInt((String)st2.nextElement());
			sum=sum+price*quantity;

		}
		double GST= sum/10.0;
		System.out.println("the GST is: "+GST);
		}

	}
```

=--

Q. to split and taking sum of tokens

```
import java.util.StringTokenizer;

public class Tokenizer {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		int sum=0;
		System.out.println("Enter the integers");
		Scanner sc=new Scanner(System.in);
		String str=sc.nextLine();
		StringTokenizer st=new StringTokenizer(str," ");
		while (st.hasMoreTokens()) {
			String temp=st.nextToken();
			int n= Integer.parseInt(temp);
			System.out.println(n);
			sum=sum+n;

		}
		System.out.println(sum);

		sc.close();
	}

}
```

=--

Comparator

- **Java program to sort the numbers according to their last digit**

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.List;

public class ComparatorDemo {
	public static void main(String[]args) {
		List<Integer>values=new ArrayList<>();
		values.add(465);
		values.add(756);
		values.add(222);
		values.add(898);

		Comparator<Integer>c=new Comparator<Integer>() {

	    public int compare(Integer i, Integer j) {
				// TODO Auto-generated method stub
				if (i%10>j%10)
					return 1;
				else
					return -1;
			}

		};
		Collections.sort(values,c);
		for (Integer o:values) {
			System.out.println(o);
		}

		}
}
```

=--

ArrayList

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Iterator;
import java.util.List;
import java.util.Scanner;

 class Student {
   private String name;
	private int rollno;
	private float marks;
	private String state;
```

```
 public Student( String name,int rollno,float marks,String state) {
		this.name=name;
		this.rollno=rollno;
		this. marks= marks;
		this.state=state;

}

public String getName() {
		return name;
}
public void setName(String name) {
		this.name = name;
}
public int getRollno() {
		return rollno;
}
public void setRollno(int rollno) {
		this.rollno = rollno;
}
public float getMarks() {
		return marks;
}
public void setMarks(float marks) {
		this.marks = marks;
}
public String getState() {
		return state;
}
public void setState(String state) {
		this.state = state;
}
public String toString() {
		return "Student [name= "+ name+ ", rollno="+rollno+",marks= "+marks+",
state="+state+"]";
}

public class SetDemo {

static void sortDataByMarks(List<Student>StList) {
		 for (int i=0;i<StList.size()-1;i++) {
				for (int j=0;j<StList.size()-i-1;i++) {
						Student firstObject=StList.get(j);
						Student secObject=StList.get(j+1);

						if( firstObject.getMarks()>secObject.getMarks()) {
								StList.set(j,secObject);
								StList.set(j+1,firstObject);
						}
				}
```

```
                }


}
static void sortDataByState(List<Student>StList) {
        for (int i=0;i<StList.size()-1;i++) {
                for (int j=0;j<StList.size()-i-1;i++) {
                        Student firstObject=StList.get(j);
                        Student secObject=StList.get(j+1);

                        if( firstObject.getState().compareTo(secObject.getState())>0){
                                StList.set(j,secObject);
                                StList.set(j+1,firstObject);
}
                }
        }
}


}
static List<Student>getStudentsArrayList(){

            Scanner sc=new Scanner(System.in);
                System.out.println("Enter the record: ");
                String StRecord=sc.next();
                String records[]=StRecord.split("#");

                List<Student>StList=new ArrayList<>();

                for(String data: records) {
                        String stData[]=data.split(":");
                        String name=stData[0];
                        int rollno=Integer.parseInt(stData[1]);
                        float marks=Float.parseFloat(stData[2]);
                        String state=stData[3];

                Student currentStudent=new Student (name,rollno,marks,state);
                StList.add(currentStudent);


                }

                sc.close();
        }
        static void display(List<Student>StList) {
                Iterator<Student>itr=StList.iterator();
                while(itr.hasNext()) {
                        System.out.println(itr.next());

        }
                System.out.println();
```

```
	}
	public static void main(System[]args) {
		List<Student>StList=getStudentsArrayList();
		display(StList);
	}
}
```
=-=-=-=-=-=-=-=-=-=-=-=-=-=-==-=-=-=-=-=-=-=-=-=-=-=--=-

Comparable

- **Java program to sort student names according to their marks**

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

class Student implements Comparable<Student>{
	private String name;
	private int rollno;
	private  float marks;
	public Student(String name,int rollno,float marks) {
		this.name=name;
		this.rollno=rollno;
		this.marks=marks;
	}
	public String toString() {
		return "Student[name= "+name+", rollno= "+rollno+", marks= "+ marks+"]";
	}

public int compareTo(Student s) {
	return marks>s.marks?1:1;
}
public class ComparableDemo {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		List<Student>studs=new ArrayList<>();
		studs.add(new Student("adithya",1,100));
		studs.add(new Student("adarsh",10,95));
		studs.add(new Student("aadi",31,80));

		Collections.sort(studs);
```

```
		for(Student s: studs) {
			System.out.println(s);
		}
		}

}
}
```

REPLACE CONSONANTS lab book

```
import java.util.*;
public class ReplaceConsonants {
	public static String alterString(String str) {
		char[] ch=str.toCharArray();

		for (int i = 0; i < ch.length; i++)  {


				if(ch[i]=='a'||ch[i]=='e'||ch[i]=='i'||ch[i]=='o'||ch[i]=='u') {
					ch[i]=ch[i];
				}
				else if(ch[i]=='z') {
					ch[i]='b';
				}
				else {
					ch[i]=(char) (ch[i]+1);
				}
		}

		 return String.valueOf(ch);

	}
		public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter the string");
		String str=sc.next();
		str.toLowerCase();
		System.out.println(ReplaceConsonants.alterString(str));
		sc.close();
	}
	}
```

Lambda Expression

Q. to calculate priceperFeet using lambda expression

```
package javaProjectExamples;

interface PaintingCost{
	public float getPaintingCost(float pricePerFeet);


	}
public class lambdaExpression {
	public static void main(String[]args) {

		PaintingCost pc=(float pricePerFeet)->{
		 return  4*3.14f*4.5f*4.5f*pricePerFeet;
		};
		System.out.println(pc.getPaintingCost(4.0f));
	}
```

2)**Q. Sort the person according to his name or weight or age**

```
import java.util.Comparator;
import java.util.Iterator;
import java.util.Set;
import java.util.TreeSet;

class  Person implements Comparator<Person>{
	private int age;
	private float weight;
	private String name;

	public Person(int age, float weight, String name) {
		super();
		this.age = age;
		this.weight = weight;
		this.name = name;
	}

	public int getAge() {
		return age;
	}

	public void setAge(int age) {
		this.age = age;
	}
```

```
	public float getWeight() {
		return weight;
	}

	public void setWeight(float weight) {
		this.weight = weight;
	}

	public String getName() {
		return name;
	}

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

	@Override
	public String toString() {
		return "Person [age=" + age + ", weight=" + weight + ", name=" + name + "]";
	}
	@Override
	public int compare(Person o1, Person o2) {
		// TODO Auto-generated method stub
		return 0;
	}

}
public class LambdaDemo {

	public static void main(String[] args) {
		Person personOne=new Person(25,68,"ABC");
		Person personTwo=new Person(26,40,"DEF");

		Person personThree=new Person(24,91,"GHI");

		// TODO Auto-generated method stub
			Comparator<Person>cmp=(p1,p2)->{
				if (p1.getAge()>p2.getAge())
					return 1;
				else if(p1.getAge()<p2.getAge())
					return -1;
				if (p1.getWeight()>p2.getWeight())
					return 1;
				else if(p1.getWeight()<p2.getWeight())
					return -1;
				return p1.getName().compareTo(p2.getName());
		};
		Set <Person>set=new TreeSet<>(cmp);
		set.add(personOne);
```

```
                set.add(personTwo);
                set.add(personThree);

Iterator<Person>itr=set.iterator();
while(itr.hasNext()) {
        System.out.println(itr);
}
        }
}
```

Employee Salary Doselect

Sort the employee according to his designation

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Iterator;




class Employee {
        private String name;
        private String designation;
        private float salary ;



        public Employee(String name,String designation,float salary) {
                this.name=name;
                this.designation=designation;
                this.salary=salary;
        }

        public String getName() {
                return name;
        }

        public void setName(String name) {
                this.name = name;
        }
```

```
	public String getDesignation() {
		return designation;
	}

	public void setDesignation(String designation) {
		this.designation = designation;
	}

public float getSalary() {
	return salary;
}
public void setSalary(float salary) {
	this.salary=salary;
}
}


class Company{
	ArrayList<Employee>el =new ArrayList<>();


ArrayList<String> uniqueDesignation(){
	ArrayList<String> d =new ArrayList<>();
	Iterator<Employee> it=el.iterator();
	while(it.hasNext()) {
		Employee value=it.next();
		if(!d.contains(value.getDesignation()))
			d.add(value.getDesignation());
	}
	Collections.sort(d);
	return d;



}

String updateSalart(String designation,float addSalary) {
	Iterator<Employee> it=el.iterator();
	while(it.hasNext()) {
		Employee cr=it.next();
		if(cr.getDesignation().equals(designation)) {
			cr.setSalary(cr.getSalary()+addSalary);
			return "Salary updated";
```

```
		}
	}
	return "no designation found";

}
}
class Check{
public static  void main (String[] args) {


	Company obj=new Company();
	obj.el.add(new Employee("Steve","Manager",20000));
	obj.el.add(new Employee("bob","Developer",15000));
	obj.el.add(new Employee("alice","Developer",15000));
	System.out.println(obj.uniqueDesignation());
	System.out.println(obj.updateSalart("Developer",500));
}
}
```

GARRY DOSELECT(String manipulation)

```
import java.util.ArrayList;
public class Source {
 public String listStartToEnd(ArrayList<String>list,int start,int end){
	String concat= " ";
	for(int i=start;i<=end;i++) {
		concat=concat+list.get(i);
	}
	return concat;
}
 public ArrayList<String> addBefore(ArrayList<String>list,String p,String q){
	 list.add(2, "Super Naturals");
	 return list;
 }

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		ArrayList<String> tvShows=new ArrayList<>();
		tvShows.add("Breaking bad");
		tvShows.add("GOT");
		tvShows.add("Friends");
		tvShows.add("Prison break");
		Source a=new Source();
		System.out.println(a.listStartToEnd(tvShows, 0, 2));
		System.out.println(a.addBefore(tvShows, "Friends", "Super natural"))
```

```
	}

}
```

MAP FILTER DO SELECT

```
import java.util.*;
import java.util.stream.Collectors;
public class User {
private String firstName;
private String lastName;
private int age;

User(String firstName,String lastName,int age){
	this.firstName=firstName;
	this.lastName=lastName;
	this.age=age;
}

	public String getFirstName() {
	return firstName;
}

public void setFirstName(String firstName) {
	this.firstName = firstName;
}

public String getLastName() {
	return lastName;
}

public void setLastName(String lastName) {
	this.lastName=lastName;
}

public int getAge() {
	return age;
}

public void setAge(int age) {
	this.age = age;
}

@Override
public String toString() {
	return  "{"+firstName + ", " + lastName + ", " + age+"}" ;
```

```
}

}
class Implementation{



	public static List<User>  filterAge(List<User> list){
		List<User>list1=list.stream()
		.filter(m->m.getAge()>40).collect(Collectors.toList());


		return list1;
	}
	public  static User findYoungest(List<User> list) {

		Optional<User>list1= list.stream()
				.min(Comparator.comparing(User::getAge));
		User u=list1.get();
		return u ;
	}
}

 class Check{
public static void main(String[] args) {
	// TODO Auto-generated method stub
	List<User>list=new ArrayList<>();
	list.add(new User("Scarlet","Jonson",25));
	list.add(new User("David","Beckham",45));
	Implementation im=new Implementation();
	System.out.println(im.filterAge(list));
	System.out.println(im.findYoungest(list));

}

}
```

Validation(Exception handling) do select

```
class InvalidNameException extends Exception {
	InvalidNameException (String s){
       super(s);
	}

}
```

```
 class WorkForce{
	String firstName;
	String lastName;
 }

class WorkForceValidation{
	public String nameValidation(WorkForce w,String firstName,String lastName) {
```



```
		try {
		if(firstName==" "||lastName==" ") {
			throw new NullPointerException("Entry Missing");
		}else if(firstName.length()==0||lastName.length()==0) {
			throw new StringIndexOutOfBoundsException("Index Out of
Bound");
		}else if(firstName.charAt(0)==0-9||lastName.charAt(0)==0-9) {
			throw new InvalidNameException("First character is invalid");
		}else {
		w.firstName=firstName;
		w.lastName = lastName;
		return w.firstName+w.lastName;
		}

	}catch(NullPointerException ne) {
		return "Entry Missing";

	}catch(StringIndexOutOfBoundsException se) {
		return "Index Out of Bound";
	}
		catch(InvalidNameException ie) {
		return "First character is invalid";
	}

	}

}


  class WorkForceDoSelect{
	public static void main(String[] args) throws InvalidNameException {
		// TODO Auto-generated method stub
		WorkForce wf=new WorkForce();
		WorkForceValidation wfv=new WorkForceValidation();
		System.out.println(wfv.nameValidation(wf,"Vivs","Daniel"));
```

```
        }
}
```

HASH SET DoSelet example

```
public class HashSetDoSelect {

public Set<Integer>subtract(Set<Integer>a,Set<Integer>b){
                Set<Integer> firstSet=new HashSet<Integer>(a);
                Set<Integer> secSet=new HashSet<Integer>(b);
                firstSet.removeAll(secSet);
                return firstSet;
        }
public Set<Integer>union(Set<Integer>a,Set<Integer>b){
        Set<Integer> firstSet=new HashSet<Integer>(a);
        Set<Integer> secSet=new HashSet<Integer>(b);
        firstSet.addAll(secSet);
        return firstSet;
}

public Set<Integer>intersect(Set<Integer>a,Set<Integer>b){


        Set<Integer> firstSet=new HashSet<Integer>(a);
        Set<Integer>secSet=new HashSet<Integer>(b);
        firstSet.retainAll(secSet);
        return firstSet;

}


        public static void main(String[] args) {
                // TODO Auto-generated method stub
    Set<Integer>set1=new HashSet<Integer>();
          set1.add(5);
          set1.add(6);
          set1.add(7);
          set1.add(8);
     Set<Integer>set2=new HashSet<Integer>();
          set1.add(9);
          set1.add(3);
          set1.add(7);

          HashSetDoSelect s=new HashSetDoSelect();
        System.out.println(s.subtract(set1,set2));
```

```
        System.out.println(s.union(set1,set2));
        System.out.println( s.intersect(set1,set2));

        }

}
```

BANDEJA PAISA DOSELECT

```
import java.util.List;
import java.util.Arrays;
import java.util.Iterator;

class Product {
private int id;
private String name;
private double price;
        public Product (int id,String name,double price) {
                this.id=id;
                this.name=name;
                this.price=price;
        }
        public int getId() {
                return id;
        }
        public void setId(int id) {
                this.id=id;
        }
        public String getName() {
                return name;
        }
        public void setName(String name) {
                this.name=name;
        }
        public double getPrice() {
                return price;
        }
        public void setPrice(double price) {
                this.price=price;
        }
        @Override
        public String toString() {
                return "Product {id=" + id + ", name=" + name + ", price=" + price + "}";
        }

 }
 class Implementation{
        public long getProductCount(List<Product>list,String productName) {
                long count=0L;
                Iterator<Product>itr=list.iterator();
```

```
		while(itr.hasNext()) {
			Product i=itr.next();
			if(i.getName().equals(productName)) {
				count++;
			}
					}
		return count;

	}
	public Product getModelDetails (List<Product>list,String productName,int id) {
		Iterator<Product>itr=list.iterator();
		while(itr.hasNext()) {
			Product i=itr.next();
			if(i.getName().equals(productName)||i.getId()==id) {
				return i;
			}


	}
		return null;
}
}
 class Check{

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Product pr1=new Product(1,"jade",44.9);
		Product pr2=new Product(2,"jane",25.50);
		Product pr3=new Product(3,"Bandeja Paisa",35.4);
		Product pr4=new Product(4,"tortilla",15.0);

		List<Product> products=Arrays.asList(pr1,pr2,pr3,pr4);
		Implementation im=new Implementation();
System.out.println(im.getProductCount(products, "tortilla"));
System.out.println(im.getModelDetails(products, "tortilla", 4));

	}

}
```

EXCEPTION HANDLING

```
class NotEligibleException extends Exception {
	NotEligibleException (String s){
       super(s);
	}
}
class companyJobRepository{
```

```
	static String getJobPrediction(int age,String highestQualification)throws
NotEligibleException {
		if (age<19){
			throw new NotEligibleException("you are underage for any job");
		}else if(age>=21 && highestQualification.equals("B.E")) {
			return "We have openings for junior developer";
		} else if(age>=26 &&
highestQualification.equals("M.S")||highestQualification.equals("PhD")) {
		   return "We have openings for senior developer";
		}else if(age>=19 &&
!(highestQualification.equals("B.E")||highestQualification.equals("PhD")||highestQualifica
tion.equals("M.S"))) {
			throw new NotEligibleException("We do not have any job that
matches your qualifications");
		}else
			return "Sorry we have no openings for now";
		}

		public String searchForJob(int age,String highestQualification)throws
NotEligibleException {
			String message="";
			if (age>=200||age<=0) {
				throw new NotEligibleException("The age entered is not
typical for a human being ");

			}try {
				message=companyJobRepository.getJobPrediction(age,
highestQualification);

			}catch(NotEligibleException ex) {
				message= ex.toString();
			}
			return message;
		}

	public static void main(String[] args)throws Exception {
		// TODO Auto-generated method stub
		Scanner sc=new Scanner(System.in);
		System.out.print("Enter the age: ");
		int age=sc.nextInt();
		System.out.print("Enter the highest Qualification: ");
		String highQ=sc.next();
		companyJobRepository c=new companyJobRepository();
		 try {

		 System.out.print(c.searchForJob(age,highQ));
```

```
                }catch(NotEligibleException ex) {
                        System.out.println(ex);

                }
                sc.close();
                }
}
```

SHIPPING

```
class SamePlaceException extends Exception{
        public SamePlaceException(String str) {
                super(str);
        }
}
class WeightException extends Exception{
        public WeightException(String str) {
                super(str);
        }
}
public class Shipping {
String sourcePlace;
String destinationPlace;
int netWeight;
int totalWeight;

        public Shipping(String sourcePlace, String destinationPlace, int netWeight, int
totalWeight) {
        super();
        this.sourcePlace = sourcePlace;
        this.destinationPlace = destinationPlace;
        this.netWeight = netWeight;
        this.totalWeight = totalWeight;
}
}
class Implementation {
        public String validator(Shipping details) throws Exception{
                if(details.sourcePlace.equals(details.destinationPlace)) {
                        System.out.println("source and destination cannot be same");
                        throw new SamePlaceException("source and destination cannot be
same");


                }

                else if (details.netWeight>details.totalWeight) {
                        System.out.println("net weight cannot be greater than total weight");
```

```
                        throw new WeightException("net weight cannot be greater than
total weight");
                }

                return null ;
        }

public float totalBill(Shipping details) {

        float totalBill;
                try {
                if(validator(details) != null ) {
                        throw new Exception();

                }

                else {

                                System.out.println("Shipping details are valid");

                         totalBill=(details.totalWeight)*5;
                         System.out.println(totalBill);
                        }
        }
        catch(SamePlaceException | WeightException ex){
                System.out.println(0.0);
                return 0.0f;

        }
        catch(Exception xe) {
                System.out.println(-1.0);
                return -1.0f;
        }

                return totalBill;
        }
        public static void main(String[] args) throws Exception {
                // TODO Auto-generated method stub
                Shipping data=new Shipping("Delhi","noida",9,10);
                Implementation imp=new Implementation();
                imp.validator(data);
                imp.totalBill(data);

        }

}
```

Video game

```
import java.util.*;

class User {
	String name;
	int balance;

	User(String name, int balance) {
		this.name = name;
		this.balance = balance;
	}

	void addBalance(int amount) {
		balance = balance + amount;
	}

	String currentBalance() {

		return "Hello " + name + " your account balance is " + balance;
	}
}

class Game {
	HashMap<String, Integer> map = new HashMap<String, Integer>();

	String playGame(String gameName, User details) {
		Set<HashMap.Entry<String, Integer>> setMap = map.entrySet();
		Iterator<HashMap.Entry<String, Integer>> itr = setMap.iterator();

		while (itr.hasNext()) {
			HashMap.Entry<String, Integer> current = itr.next();
			if (current.getKey().equals(gameName)) {
				details.balance = details.balance - current.getValue();
				return "Hello " + details.name + ", thanks for playing " +
gameName + " and your current balance is "
						+ details.balance;
			}
		}

		return "Game not found";
	}

	void addGame(String gameName, int gamePrice) {
		map.put(gameName, gamePrice);
```

```
	}

}

public class VideoGame {

	public static void main(String[] args) {
		User user1 = new User("Shalini", 200);
		User user2 = new User("Avila", 300);
		System.out.println(user1.currentBalance());
		System.out.println(user2.currentBalance());

		System.out.println();
		user1.addBalance(50);
		user2.addBalance(40);
		System.out.println(user1.currentBalance());// 250
		System.out.println(user2.currentBalance());// 340

		Game game = new Game();
		game.addGame("Ludo", 50);
		game.addGame("Chess", 30);
		System.out.println();

		System.out.println(game.playGame("Chess", user1));
		System.out.println(game.playGame("Ludo", user2));

	}

}
```

SPEED VIDEOGAME EXCEPTION

```
class SpeedInvalidException extends Exception{
	public SpeedInvalidException(String s) {
		super(s);
	}
}

class Speed{
	String  speed;
	}

class SpeedImplementation {
	public String setSpeed(Speed s,int spd) {
		try {
		if(spd<30 || spd>90) {
```

```
                    throw new SpeedInvalidException("SpeedInvalidException");
                }else {
                s.speed="Valid Speed";}
        }catch(SpeedInvalidException se) {
        s.speed="Invalid Speed";
        return "SpeedInvalidException";
        }
        return s.speed;
        }

}
 class Main {
        public static void main(String[] args)throws Exception {
                // TODO Auto-generated method stub

Speed s=new Speed();
SpeedImplementation si=new SpeedImplementation();
System.out.println(si.setSpeed(s,60));
}
}
```

STRING MANIPULATION DOSELECT

```
public class BMI {
public float returnWeight(String str) {
        str=str.replace('-', '.');
        String [] arrstr=str.split("@");
 String weight=arrstr[0];

          System.out.println("Weight="+weight);

        return 0.0f;
}
public float returnHeight(String str) {
        str=str.replace('-','.');
        String [] arrstr=str.split("@");
        String height=arrstr[1];
        System.out.println("Height="+height);
        return 0.0f;
}
        public static void main(String[] args) {
                // TODO Auto-generated method stub
```

```
		String str="68-45@1-78";

		BMI b=new BMI();
		b.returnWeight(str);
		b.returnHeight(str);

	}

}
```

COLLECTION HASHMAP DO SELECT

```
package javaProjectExamples;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.Iterator;
import java.util.List;
import java.util.Map;

class Us{
	String name;
	int balance;
	public Us(String name, int balance) {
		super();
		this.name = name;
		this.balance = balance;
	}
	int sum=0;
 void addBalance(int amount) {

		sum=amount+balance;

	}

 String currentBalance() {
	return "\"Hello " + name + " your current balance is " + sum + ".\"";
}
}
```

```
class Game{
	HashMap<String,Integer>map=new HashMap<String,Integer>();
	public  void addGame(String gameName,int gamePrice) {
		map.put(gameName,gamePrice);
	}

	public String playGame(String gameName,Us details) {
		if(map.containsKey(gameName)) {
			Iterator<Map.Entry<String,Integer>> iter=map.entrySet().iterator();
			while(iter.hasNext()) {
				Map.Entry<String,Integer> entry= iter.next();
				if(entry.getKey().equals(gameName)) {
					int price=entry.getValue();
					details.sum=details.sum-price;
				}
				return "\"" + details.name + ", thanks for playing " +
gameName +   " and your current balance is "  + details.sum +"\"";
	}

		}
		return gameName;

	}


public class VideoGameDoSelect {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Us u=new Us("Steve", 500);
		u.addBalance(500);
		Game play=new Game();
		play.addGame("Ludo",20);
		play.addGame("Chess",10);
		play.addGame("Hangman",30);
		System.out.println(play.playGame("Ludo",u));
		System.out.println(u.currentBalance());


	}
```

```
}
}
```

Exception handling and regex

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;

class customer{
		String name;
		String mobilenum;
		String custmerId;
		public customer(String name, String mobilenum, String custmerId) {
			super();
			this.name = name;
			this.mobilenum = mobilenum;
			this.custmerId = custmerId;
		}
}
class Validator{
	public String validateCustomerID(customer c)throws Exception{
		String result="";
		String num=c.mobilenum;
		String fourDigit=num.substring(0,4);
		String name=c.name;
		String newname=name.substring(name.length()-2);
		result=fourDigit.concat(newname);
		if (!(result.matches(c.custmerId))){
		  throw new InvalidCustomerIDException("Invalid customerID");
		}else
	return "valid Cid";

	}
	public  String validateMobileNo(customer c)throws Exception{
		String mobnum=c.mobilenum;
		long num=Long.parseLong(c.mobilenum);
		if (!(c.mobilenum.matches("[6-9][0-9]{9}"))){
			throw new InvalidMobileNoException("Invalid MObile number");
		}else
			return "Valid Mobile number";
```

```
	}

	class InvalidCustomerIDException extends Exception {
		public InvalidCustomerIDException (String str) {
			super(str);
		}
	}
		class InvalidMobileNoException  extends Exception{
			public InvalidMobileNoException (String str){
				super(str);
			}
		}

public class CustomerCareException {

	public static void main(String[] args)throws Exception {
		// TODO Auto-generated method stub
customer obj=new customer("Steve","9898111111","9898ve");
Validator val=new Validator();
String CID=val.validateCustomerID(obj);
String mob=val.validateMobileNo(obj);
System.out.println(mob);
System.out.println(CID);


	}
}
}
```

Regex

```
public class InputValidationUsingRegex {
   public static void main(String[] args) {
      //create a pattern for mobile number
      String inputPatternForMobile = "^[6-9]\\d{9}";

      //create reference variable of pattern
      Pattern pattern = null;
      Matcher matcher = null;
```

```
    String validMobileNumber = "9865321540";
    pattern = Pattern.compile(inputPatternForMobile);
    matcher = pattern.matcher(validMobileNumber);
    System.out.println("9865321540 is valid mobile number? " + matcher.matches());

    String invalidMobileNumber = "5896325412";
    pattern = Pattern.compile(inputPatternForMobile);
    matcher = pattern.matcher(invalidMobileNumber);
    System.out.println("5896325412 is valid mobile number? " + matcher.matches());


    System.out.println();

    String inputPatternForName = "^[A-Za-z\\s]+$";

    String validName = "Ajay Khanna";
    pattern = Pattern.compile(inputPatternForName);
    matcher = pattern.matcher(validName);
    System.out.println("Ajay Khanna is valid name? " + matcher.matches());


    String inValidName = "$123";
    pattern = Pattern.compile(inputPatternForName);
    matcher = pattern.matcher(inValidName);
    System.out.println("$123 is valid name? " + matcher.matches());

    System.out.println();

    String inputPatternForEmail = "^[A-Za-z0-9_.]+@[A-Za-z0-9_.]+$";
    String validEmail = "kapil@capgemini.com";
    pattern = Pattern.compile(inputPatternForEmail);
    matcher = pattern.matcher(validEmail);
    System.out.println("kapil@capgemini.com is valid name? " + matcher.matches());

    String inValidEmail = "kapil";
    pattern = Pattern.compile(inputPatternForEmail);
    matcher = pattern.matcher(inValidEmail);
    System.out.println("kapil is valid name? " + matcher.matches());
}
```

}

VIDEO GAMECOLLECTIONS DOSELECT

```
import java.util.*;

class User {
	String name;
	int balance;

	User(String name, int balance) {
		this.name = name;
		this.balance = balance;
	}

	void addBalance(int amount) {
		balance = balance + amount;
	}

	String currentBalance() {

		return "Hello " + name + " your account balance is " + balance;
	}
}

class Game {
	HashMap<String, Integer> map = new HashMap<String, Integer>();

	String playGame(String gameName, User details) {
		Set<HashMap.Entry<String, Integer>> setMap = map.entrySet();
		Iterator<HashMap.Entry<String, Integer>> itr = setMap.iterator();

		while (itr.hasNext()) {
			HashMap.Entry<String, Integer> current = itr.next();
			if (current.getKey().equals(gameName)) {
				details.balance = details.balance - current.getValue();
```

```
                                return "Hello " + details.name + ", thanks for playing " +
gameName + " and your current balance is "
                                                + details.balance;
                        }
                }

                return "Game not found";
        }

        void addGame(String gameName, int gamePrice) {
                map.put(gameName, gamePrice);

        }

}

public class VideoGame {

        public static void main(String[] args) {
                User user1 = new User("Sini", 200);
                User user2 = new User("Pal", 300);
                System.out.println(user1.currentBalance());
                System.out.println(user2.currentBalance());

                System.out.println();
                user1.addBalance(50);
                user2.addBalance(40);
                System.out.println(user1.currentBalance());// 250
                System.out.println(user2.currentBalance());// 340

                Game game = new Game();
                game.addGame("Ludo", 50);
                game.addGame("Chess", 30);
                System.out.println();

                System.out.println(game.playGame("Chess", user1));
                System.out.println(game.playGame("Ludo", user2));

        }
```

```
}
```