***Dt : 16/11/2022***

***EmployeeValues.java***

***package test;***

***public class EmployeeValues extends Object{***

***public String name,desg;***

***public int bSal;***

***public float totSal;***

***public EmployeeValues(String name,String desg,int bSal,float totSal) {***

***this.name=name;***

***this.desg=desg;***

***this.bSal=bSal;***

***this.totSal=totSal;***

***}***

***@Override***

***public String toString() {***

***return name+"\t"+desg+"\t"+bSal+"\t"+totSal;***

***}***

***}***

***Ex : DemoMap.java***

***package maccess;***

***import java.util.\*;***

***import test.\*;***

***public class DemoMap {***

***public static void main(String[] args) {***

***Map<String,EmployeeValues> ob = null;***

***String name=null;***

***Scanner s = new Scanner(System.in);***

***try(s;){***

***try {***

***while(true) {***

***System.out.println("\*\*\*\*\*Choice\*\*\*\*\*");***

***System.out.println("1.HashMap\n2.LinkedHasMap\n3.TreeMap\n4.Hashtable\n5.exit");***

***System.out.println("Enter the Choice:");***

***switch(Integer.parseInt(s.nextLine())) {***

***case 1:***

***ob = new HashMap<String,EmployeeValues>();***

***name="HashMap";***

***break;***

***case 2:***

***ob = new LinkedHashMap<String,EmployeeValues>();***

***name="LinkedHashMap";***

***break;***

***case 3:***

***ob = new TreeMap<String,EmployeeValues>();***

***name="TreeMap";***

***break;***

***case 4:***

***ob = new Hashtable<String,EmployeeValues>();***

***name="Hashtable";***

***break;***

***case 5:***

***System.out.println("Operation stopped on Map...");***

***System.exit(0);***

***default:***

***System.out.println("Invalid Choice...");***

***}//end of switch***

***System.out.println("Perform operations on "+name);***

***xyz:***

***while(true) {***

***System.out.println("====Choice====");***

***System.out.println("1.put(K,V)\n2.remove(object)\n3.get(object)\n4.keySet()\n5.values()\n6.display\n7.exit");***

***System.out.println("Enter the Choice:");***

***switch(Integer.parseInt(s.nextLine())) {***

***case 1:***

***System.out.println("Enter EmpId:");***

***String eId = s.nextLine();***

***System.out.println("Enter EmpName:");***

***String eName = s.nextLine();***

***System.out.println("Enter Desg:");***

***String desg = s.nextLine();***

***System.out.println("Enter the bSal:");***

***int bSal = Integer.parseInt(s.nextLine());***

***float totSal = bSal+(0.93F\*bSal)+(0.63F\*bSal);***

***ob.put(new String(eId),***

***new EmployeeValues(eName,desg,bSal,totSal));***

***System.out.println("details added to Map....");***

***break;***

***case 2:***

***if(ob.isEmpty()) {***

***System.out.println("Map is empty....");***

***}else {***

***System.out.println("Enter EmpId:");***

***String id = new String(s.nextLine());***

***ob.remove(id);***

***System.out.println("Object removed...");***

***ob.forEach((p,q)->***

***{***

***System.out.println(p+"\t"+q);***

***});***

***}***

***break;***

***case 3:***

***if(ob.isEmpty()) {***

***System.out.println("Map is empty....");***

***}else {***

***System.out.println("Enter EmpId:");***

***String id2 = new String(s.nextLine());***

***EmployeeValues ev = ob.get(id2);***

***System.out.println(ev.toString());***

***}***

***break;***

***case 4:***

***if(ob.isEmpty()) {***

***System.out.println("Map is empty....");***

***}else {***

***Set<String> ob2 = ob.keySet();***

***ob2.forEach((x)->***

***{***

***System.out.println(x.toString());***

***});***

***}***

***break;***

***case 5:***

***if(ob.isEmpty()) {***

***System.out.println("Map is empty....");***

***}else {***

***Collection<EmployeeValues> ob3 = ob.values();***

***ob3.forEach((y)->***

***{***

***System.out.println(y.toString());***

***});***

***}***

***break;***

***case 6:***

***if(ob.isEmpty()) {***

***System.out.println("Map is empty....");***

***}else {***

***ob.forEach((p,q)->***

***{***

***System.out.println(p+"\t"+q);***

***});***

***}***

***break;***

***case 7:***

***System.out.println("Operations stopped on "+name);***

***break xyz;***

***default:***

***System.out.println("Invalid Choice...");***

***}//end of switch***

***}//end of loop***

***}//end of while***

***}catch(Exception e) {e.printStackTrace();}***

***}//end of try***

***}***

***}***

***o/p:***

***\*\*\*\*\*Choice\*\*\*\*\****

***1.HashMap***

***2.LinkedHasMap***

***3.TreeMap***

***4.Hashtable***

***5.exit***

***Enter the Choice:***

***3***

***Perform operations on TreeMap***

***====Choice====***

***1.put(K,V)***

***2.remove(object)***

***3.get(object)***

***4.keySet()***

***5.values()***

***6.display***

***7.exit***

***Enter the Choice:***

***1***

***Enter EmpId:***

***A111***

***Enter EmpName:***

***Raj***

***Enter Desg:***

***SE***

***Enter the bSal:***

***17000***

***details added to Map....***

***====Choice====***

***1.put(K,V)***

***2.remove(object)***

***3.get(object)***

***4.keySet()***

***5.values()***

***6.display***

***7.exit***

***Enter the Choice:***

***1***

***Enter EmpId:***

***A001***

***Enter EmpName:***

***Ram***

***Enter Desg:***

***TE***

***Enter the bSal:***

***18000***

***details added to Map....***

***====Choice====***

***1.put(K,V)***

***2.remove(object)***

***3.get(object)***

***4.keySet()***

***5.values()***

***6.display***

***7.exit***

***Enter the Choice:***

***1***

***Enter EmpId:***

***A11***

***Enter EmpName:***

***Alex***

***Enter Desg:***

***TE***

***Enter the bSal:***

***19000***

***details added to Map....***

***====Choice====***

***1.put(K,V)***

***2.remove(object)***

***3.get(object)***

***4.keySet()***

***5.values()***

***6.display***

***7.exit***

***Enter the Choice:***

***6***

***A001 Ram TE 18000 46080.0***

***A11 Alex TE 19000 48640.0***

***A111 Raj SE 17000 43520.0***

***====Choice====***

***1.put(K,V)***

***2.remove(object)***

***3.get(object)***

***4.keySet()***

***5.values()***

***6.display***

***7.exit***

***Enter the Choice:***

***7***

***Operations stopped on TreeMap***

***\*\*\*\*\*Choice\*\*\*\*\****

***1.HashMap***

***2.LinkedHasMap***

***3.TreeMap***

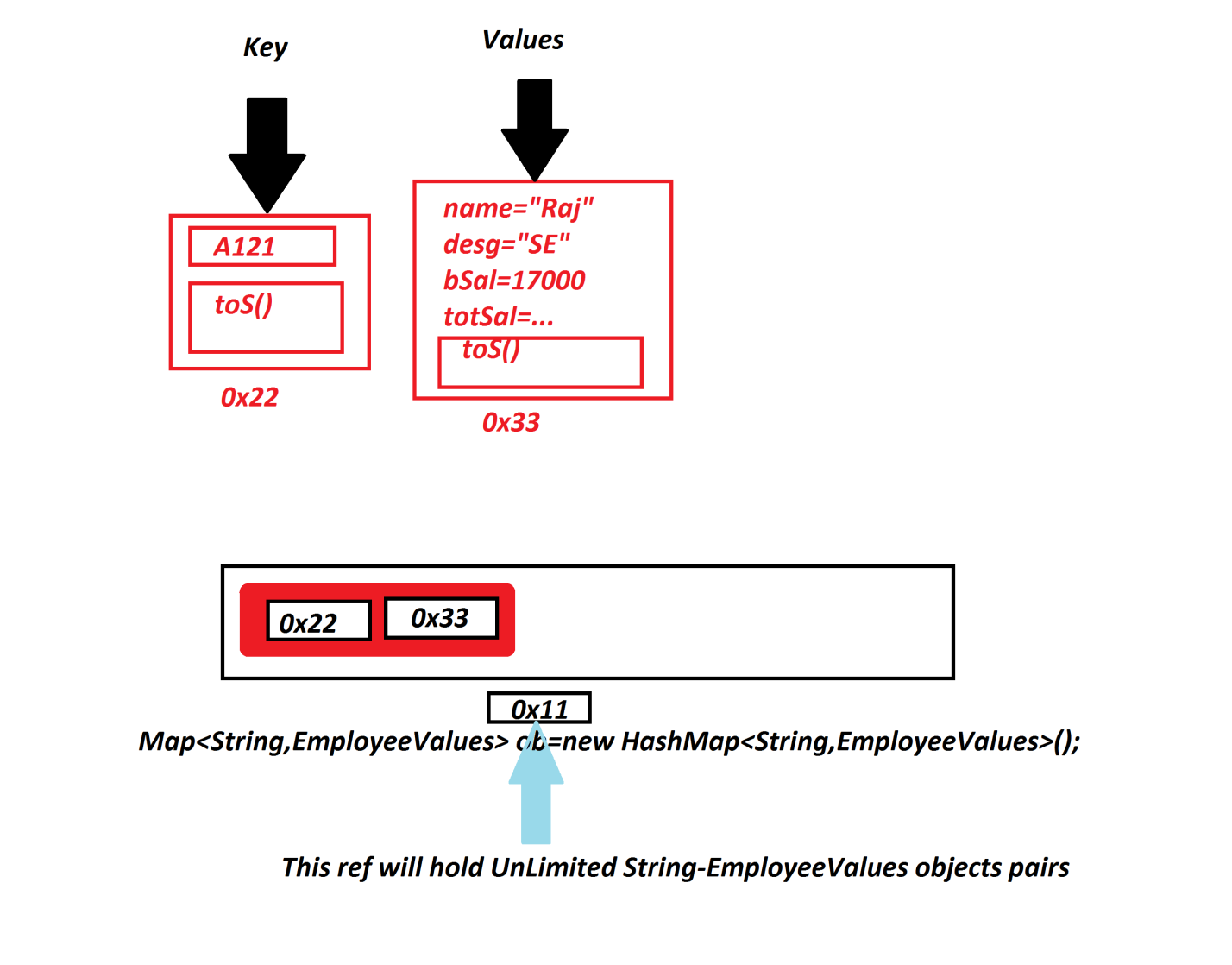
***4.Hashtable***

***5.exit***

***Enter the Choice:***

***5***

***Operation stopped on Map...***

******

***==========================================================================***

***\*imp***

***Cursor Statements in JCF:***

***=>The statements which are used to retrieve elements from Collection<E> objects***

***are known as Cursor Statements.***

***=>The following are some important Cursor Statements in JCF:***

***1.Iterator<E>***

***2.ListIterator<T>***

***3.Enumeration<E>***

***4.Spliterator<T>***

***1.Iterator<E>:***

***=>Iterator<E> is an interface from java.util package and which is used to retrieve***

***elements from Collection<E> objects in forward direction.***

***syntax:***

***Iterator<E> it = ob.iterator();***

***2.ListIterator<T>:***

***=>ListIterator<E> is an Child-Interface of Iterator<E> and which is used to***

***retrieve elements from List<E> Objects in both directions forward and backward.***

***=>The following are some important methods of ListIterator:***

***public abstract boolean hasNext();***

***public abstract E next();***

***public abstract boolean hasPrevious();***

***public abstract E previous();***

***syntax:***

***ListIterator<Integer> li = v.listIterator();***

***Ex : CursorStatements.java***

***package maccess;***

***import java.util.\*;***

***public class CursorStatements {***

***@SuppressWarnings("removal")***

***public static void main(String[] args) {***

***Vector<Integer> v = new Vector<Integer>();***

***for(int i=1;i<=5;i++) {***

***v.add(new ~~Integer~~(i));***

***}//end of loop***

***System.out.println("\*\*\*\*ListLiterator<E>\*\*\*\*");***

***ListIterator<Integer> li = v.listIterator();***

***System.out.print("Forward : ");***

***while(li.hasNext()) {***

***System.out.print(li.next()+" ");***

***}//end of loop***

***System.out.print("\nBackward : ");***

***while(li.hasPrevious()) {***

***System.out.print(li.previous()+" ");***

***}//end of loop***

***System.out.println("\n\*\*\*\*Enumeration<E>\*\*\*\*");***

***Enumeration<Integer> e = v.elements();***

***while(e.hasMoreElements()) {***

***System.out.print(e.nextElement()+" ");***

***}//end of loop***

***}***

***}***

***o/p:***

***\*\*\*\*ListLiterator<E>\*\*\*\****

***Forward : 1 2 3 4 5***

***Backward : 5 4 3 2 1***

***\*\*\*\*Enumeration<E>\*\*\*\****

***1 2 3 4 5***