# Array List

# Java ArrayList class uses a dynamic array for storing the elements.

# • Java ArrayList class can contain duplicate elements.

# • Java ArrayList class maintains insertion order.

# • Java ArrayList class is non synchronized.

# • Java ArrayList allows random access because array works at the index basis.

# • In Java ArrayList class, manipulation is slow because a lot of shifting needs

# to be occurred if any element is removed from the array list.

# ArrayList<Student> al1 = new ArrayList<Student>();

# al1.add(s1);

# boolean add(Object o)

# void add(int index, Object element)

# boolean addAll(int index, Collection c)

# boolean addAll(Collection c)

# void clear()

# int lastIndexOf(Object o)

# Object[] toArray()

# Object[] toArray(Object[] a)

# Object clone()

# int indexOf(Object o)

# void trimToSize()

# import java.util.\*;

# class Student {

# int rollno;

# String name;

# int age;

# Student(int rollno, String name, int age) {

# this.rollno = rollno;

# this.name = name;

# this.age = age;

# }

# }

# public class CollectionList {

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

# //Creating user-defined class objects

# Student s1 = new Student(101, "Sonoo", 23);

# Student s2 = new Student(102, "Ravi", 21);

# //creating arraylist al1

# ArrayList<Student> al1 = new ArrayList<Student>();

# // adding Student class object

# al1.add(s1);

# al1.add(s2);

# // remove an item

# al1.remove(s1);

# //Creating user-defined class objects

# Student s3 = new Student(103, "Sam", 33);

# Student s4 = new Student(104, "Rob", 41);

# //creating arraylist al2

# ArrayList<Student> al2 = new ArrayList<Student>();

# al2.add(s3);

# al2.add(s4);

# // adding second list in first list

# al1.addAll(al2);

# //traversing elements of ArrayList object

# for (Student st : al1)

# System.out.println(st.rollno + " " + st.name + " " + st.age);

# al1.removeAll(al2);

# for (Student st : al1)

# System.out.println(st.rollno + " " + st.name + " " + st.age);

# }

# }

# Hash Set

## Input

import java.util.\*;

class TestCollection9{

public static void main(String args[]){

//Creating HashSet and adding elements

HashSet<String> set=new HashSet<String>();

set.add("Ravi");

set.add("Vijay");

set.add("Ravi");

set.add("Ajay");

//Traversing elements

Iterator<String> itr=set.iterator();

while(itr.hasNext()){

System.out.println(itr.next());

}

}

}

Example #2

import java.util.\*;

class Book {

int id;

String name,author,publisher;

int quantity;

public Book(int id, String name, String author, String publisher, int quantity) {

this.id = id;

this.name = name;

this.author = author;

this.publisher = publisher;

this.quantity = quantity;

}

}

public class HashSetExample {

public static void main(String[] args) {

HashSet<Book> set=new HashSet<Book>();

//Creating Books

Book b1=new Book(101,"Let us C","Yashwant Kanetkar","BPB",8);

Book b2=new Book(102,"Data Communications & Networking","Forouzan","Mc Graw Hill",4);

Book b3=new Book(103,"Operating System","Galvin","Wiley",6);

//Adding Books to HashSet

set.add(b1);

set.add(b2);

set.add(b3);

//Traversing HashSet

for(Book b:set){

System.out.println(b.id+" "+b.name+" "+b.author+" "+b.publisher+" "+b.quantity);

}

}

}

# HashMap

Difference between HashSet and HashMap

HashSet contains only values whereas HashMap contains entry(key and value).

**static int** sockMerchant(**int** arraySize, **int**[] array) {  
 HashMap <Integer, Integer> colors = **new** HashMap<Integer, Integer>();  
  
 **for** (**int** color : array) {  
 Integer frequency = colors.get(color);  
 **if** (frequency == **null**) {  
 colors.put(color, 1);  
 } **else** {  
 colors.put(color, frequency + 1);  
 }  
 }  
  
 **int** pairs = 0;  
  
 **for** (Integer frequency : colors.values()) {  
 pairs = pairs + (frequency / 2);  
 }  
  
 **return** pairs;  
}

## Input

import java.util.\*;

class TestCollection13{

public static void main(String args[]){

HashMap<Integer,String> hm = new HashMap<Integer,String>();

hm.put(100,"Amit");

hm.put(101,"Vijay");

hm.put(102,"Rahul");

for(Map.Entry m:hm.entrySet()){

System.out.println(m.getKey()+" " + m.getValue());

}

}

}

## Remove

import java.util.\*;

public class HashMapExample {

public static void main(String args[]) {

// create and populate hash map

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

map.put(101,"Let us C");

map.put(102, "Operating System");

map.put(103, "Data Communication and Networking");

System.out.println("Values before remove: "+ map);

// Remove value for key 102

map.remove(102);

System.out.println("Values after remove: "+ map);

}

}