Collection framework programs:

LISTDEMO

import java.util.\*;

class ListDemo{

public static void main(String args[]){

List<Integer> l=new ArrayList<Integer>();

l.add(10);

l.add(10);

l.add("sudheer");

for(int i=0;i<10;i++)

l.add(i);

//Iterating entire list

Iterator i=l.iterator();

while(i.hasNext()){

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

}

//remove element from List

l.remove("sudheer");

//get element at specified index

System.out.println("at index 1:"+l.get(1));

//get index of specified element of list

System.out.println("index of 3:"+l.indexOf(3));

//ForEach Iterator or method

//l.forEach(n->System.out.println(n));

}

}

SET DEMO

import java.util.\*;

class HashSetDemo{

public static void main(String args[]){

Set l=new HashSet();

l.add(10);

//l.add("sudheer");

l.add(10);

for(int i=0;i<10;i++)

l.add(i);

//Iterating entire list

Iterator i=l.iterator();

while(i.hasNext()){

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

}

//remove element from List

l.remove(1);

//get element at specified index

System.out.println("at index 1:"+l.contains(1));

//get index of specified element of list

System.out.println("size of set:"+l.size());

//ForEach Iterator or method

//l.forEach(n->System.out.println(n));

}

}

HASH MAP DEMO

import java.util.\*;

class HashMapDemo{

public static void main(String args[]){

Map<Integer,String> m=new HashMap<Integer,String>();

m.put(101,"sudheer");

m.put(104,"Siva");

m.put(103,"Pavan");

m.put("Ramesh",102);

m.forEach((key,value)->

System.out.println(key+"\t"+value));

System.out.println(m.get(101));

m.remove(101);

m.forEach((key,value)->

System.out.println(key+"\t"+value));

}

}

Functional Interface lamda expression

interface Mynum{

double getValue();

}

class FunctionalInterface{

public static void main(String args[]){

Mynum num;

num=()->10.123;

System.out.println(num.getValue());

}

}

COMPARABLE COMPARATOR Example

class Student implements Comparable<Student>{

int age;

String name;

Student(int age,String name){

this.age=age;

this.name=name;

}

public int compareTo(Student i){

if(this.age>i.age)

return 1;

else

return -1;

}

@Override

public String toString() {

return "Student{" +

"age=" + age +

", name=" + name +

'}';

}

}

Program 2

import java.util.\*;

class StudentCollection{

public static void main(String args[]){

Student s1=new Student(21,"Alex");

Student s2=new Student(15,"James");

Student s3=new Student(42,"Toe");

Student s4=new Student(25,"Bell");

List l=new ArrayList();

l.add(s1);

l.add(s2);

l.add(s3);

l.add(s4);

//AgeComparator age=new AgeComparator();

//NameComparator name=new NameComparator();

Collections.sort(l);

l.forEach(x->System.out.println(x));

}

}

Program 3

import java.util.\*;

class AgeComparator implements Comparator<Student>{

public int compare(Student i1,Student i2){

if(i1.age>i2.age)

return 1;

else

return -1;

}

}

Program 4

import java.util.\*;

class NameComparator implements Comparator<Student>{

public int compare(Student i1,Student i2){

return i1.name.compareTo(i2.name);

}

}

Wrapper class Example

public class WrapperExample {

public static void main(String[] args) {

// Primitive data types

int a = 5;

double b = 5.65;

// Converting primitives into objects using wrapper classes

Integer aObj = Integer.valueOf(a);

Double bObj = Double.valueOf(b);

// Autoboxing: Automatically converting primitive to object

Integer aAutoBox = a;

Double bAutoBox = b;

// Displaying object values

System.out.println("Integer object aObj: " + aObj);

System.out.println("Double object bObj: " + bObj);

System.out.println("Autoboxed Integer aAutoBox: " + aAutoBox);

System.out.println("Autoboxed Double bAutoBox: " + bAutoBox);

// Unboxing: Converting objects back to primitives

int aUnbox = aObj.intValue();

double bUnbox = bObj.doubleValue();

// Auto-unboxing: Automatically converting object to primitive

int aAutoUnbox = aObj;

double bAutoUnbox = bObj;

// Displaying primitive values

System.out.println("Unboxed int aUnbox: " + aUnbox);

System.out.println("Unboxed double bUnbox: " + bUnbox);

System.out.println("Auto-unboxed int aAutoUnbox: " + aAutoUnbox);

System.out.println("Auto-unboxed double bAutoUnbox: " + bAutoUnbox);

// Using some useful methods from wrapper classes

String intStr = "123";

String doubleStr = "45.67";

// Parsing strings into primitive data types

int parsedInt = Integer.parseInt(intStr);

double parsedDouble = Double.parseDouble(doubleStr);

System.out.println("Parsed int from String: " + parsedInt);

System.out.println("Parsed double from String: " + parsedDouble);

// Converting a primitive type to String

String intToStr = Integer.toString(a);

String doubleToStr = Double.toString(b);

System.out.println("Integer to String: " + intToStr);

System.out.println("Double to String: " + doubleToStr);

}

}

Generic program Example

class Gen<S>{

S ob;

Gen(S o){

ob=o;

}

S getob(){

return ob;

}

void showType(){

System.out.println("Type of S"+ob.getClass().getName());

}

}

class GenDemo{

public static void main(String args[]){

Gen<Integer> iob=new Gen<Integer>(88);

Gen<String> iob2=new Gen<String>("sudheer");

iob.showType();

iob2.showType();

}

}

Generic method example

public class GenericMethodExample {

// Generic method to print elements of an array

public static <T> void printArray(T[] array) {

for (T element : array) {

System.out.print(element + " ");

}

System.out.println();

}

public static void main(String[] args) {

// Creating arrays of different types

Integer[] intArray = {1, 2, 3, 4, 5};

Double[] doubleArray = {1.1, 2.2, 3.3, 4.4, 5.5};

Character[] charArray = {'H', 'E', 'L', 'L', 'O'};

String[] stringArray = {"Java", "Generics", "Are", "Cool"};

// Calling the generic method with different types of arrays

System.out.println("Integer Array:");

printArray(intArray);

System.out.println("Double Array:");

printArray(doubleArray);

System.out.println("Character Array:");

printArray(charArray);

System.out.println("String Array:");

printArray(stringArray);

}

}