ASSIGNMENT

1.Use a Hashset to hold employee objects. Upon running application, the details of the employees added to the HashSet should be displayed.

| **Employee (class)** |
| --- |
| Id |
| Name |
| Salary |
| Department |
| displayDetails() |

Feel free to add properties and methods to employee class

Note - If we try to store any object other than employee object in hashset, we should not be allowed to.

ANS:

**package** generics;

**import** java.util.HashSet;

**public** **class** Employee {

**int** id;

String name;

**double** salary;

String department;

**public** **void** displayDetails(**int** id, String name, **double** salary, String department) {

**this**.id = id;

**this**.name = name;

**this**.salary = salary;

**this**.department = department;

System.***out***.println(

"id= " + id + " " + "name= " + name + " " + "salary = " + salary + " department = " + department);

}

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

// **TODO** Auto generated method stub

HashSet<Employee> set = **new** HashSet<Employee>();

Employee e = **new** Employee();

// e.displayDetails(1,"sarika",253458.0,"FUll stack developer");

set.add(e);

**for** (Employee e1 : set) {

e.displayDetails(1, "sarika", 253458.0, "FUll stack developer");

}

}

}

OUTPUT: -

id= 1 name= sarika salary = 253458.0 department = FUllstack developer

2 Write an application to hold 10 random int values as key and random double values as values for HashMap. Print the data store in HashMap.

ANS: **package** generics;

**import** java.util.ArrayList;

**import** java.util.HashMap;

**import** java.util.List;

**import** java.util.Random;

**public** **class** AddRandomNo {

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

// **TODO** Auto-generated method stub

HashMap<Integer, Double> hm= **new** HashMap<>();

hm.put(1, 1.1);

hm.put(10, 10.1);

hm.put(11, 11.1);

hm.put(12, 12.1);

hm.put(13, 13.1);

hm.put(14, 14.1);

hm.put(15, 15.1);

hm.put(16, 16.1);

hm.put(17, 17.1);

hm.put(18, 18.1);

System.***out***.println(hm);

}

}

OUTPUT: {16=16.1, 1=1.1, 17=17.1, 18=18.1, 10=10.1, 11=11.1, 12=12.1, 13=13.1, 14=14.1, 15=15.1}

3. Write a generic method to exchange the positions of two different elements in an array.

Ans: **package** generics;

**import** java.util.Arrays;

**public** **class** Swapping {

// **TODO** Auto-generated method stub

**public** **static** <T> **void** swap(T[] nums, **int** left, **int** right) {

T temp = nums[right];

nums[right] = nums[left];

nums[left] = temp;

}

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

Integer[] num = { 5, 12, 3, 7, 2 };

System.***out***.println("Before shorting = "+Arrays.*toString*(num));

*swap*(num, 0, 1);

System.***out***.println("after shorting = "+Arrays.*toString*(num));

}

}

OUTPUT: Before shorting = [5, 12, 3, 7, 2]

after shorting = [12, 5, 3, 7, 2]

4. Design a class named Pair which has two properties. The name of the first property is key and that of the second property Is value. When designing the class take case of the following scenarios:

* Create an Object of Pair class to store String value for the property key and String value for the property value. Restriction Apart from String type no other types should be acceptable as Key or value Input
* e.g.

myObj.setkey("1"); myObj.setValue("Hello");

**ANS: package** generics;

**public** **class** Pair<K, V> {

**private** K key;

**private** V value;

**public** Pair(K key, V value) {

**this**.key = key;

**this**.value = value;

System.***out***.println("key = " + key + " value = " + value);

}

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

Pair<String, String> pr = **new** Pair<>("1", "sarika");

}

}

OUTPUT: key = 1 value = sarika

Explanation: if we write Pair<int, String> pr = **new** Pair<>(1, "sarika");

Then it will show error because it only contain non-primitive data type