Uttara InfoSolutions

www.uttarainfo.com

Collections Practicals 1

Note: Ex code has been given to you. You can go through that first to understand the way collections methods work and then start with this lab document.

1) WAP to compare 2 strings by == and .equals() and test whether Strings .equals() works correctly. Which equals() implementation is getting picked up?

2) Code a Person class. Create 2 instance variables, one String name and one int height. Create a parameterized constructor. Create a tester class, create 2 person objects and compare the two using object identity check and object equality checks:

```
Person p1 = new Person("Ramu",20);

Person p2 = new Person("Ramu",20);

System.out.println("identity check = "+(p1==p2));

System.out.println("equality = "+(p1.equals(p2)));
```

Check what is printed and why? Now in Person class, override equals() as discussed. Check the test class execution and observe the output.

Now in main(), print the reference of p1 and see what is shown to the monitor as output. Then override toString() in Person. Re-execute the main() and see if the printing of reference has changed.

How to override equals()?

```
public boolean equals(Object o)

{

        // do instanceof check for checking if o is a Person, else return false

        // Create new Person reference and point to o by down casting

        // Compare the state of this person object with passed person object.
        // If the state is given by primitives, use ==, if it is reference,

        //use .equals().

}
```

Have you understood why and how we should override equals()?

How to override toString()

```
public String toString()

{

        return “Person:”+name+”,”+age;

}
```

3) (See TestCollMethods.java for reference) Create a TestCollections class with main(). Create an ArrayList object like this:

```
Collection col = new ArrayList(); //import package java.util.*;
```

now invoke basic methods on col collection to test adding, searching, removing, getting the size, iteration and printing the contents. First add 5-6 strings in ArrayList. Check if add is successful. Add duplicates. SOP the collection ref. Check whether duplicates are allowed. Create a new String object with same state as another and check if contains works on collection. Add many duplicates and try to remove all occurrences.

Methods to be used and tested are:

Collection

- add(Object o)

- contains(Object o)
- size()
- remove(Object o)
- clear()
- isEmpty()
- addAll(Collection c)
- removeAll(Collection c)
- retainAll(Collection c)

```
//to iterate and print contents
for(Object o : col)
{
        System.out.println(o);
}
System.out.println(col); //directly print contents without iterating!
```

Change ArrayList to LinkedList and see if you get any difference in the methods working. Then change it to HashSet and check.

Create 2 collections with strings and then invoke addAll(), retainAll() and removeAll() and verify if you understand how the methods work.

4) After checking how collections work with strings, create Person objects and add them to a new ArrayList. Verify if addition, search, removal works correctly. Put SOPs in equals() in Person class and verify if it is being called when you search/remove.

5) Create an Employee class. An Employee has a name, email, dob, home address and office address. An Address has city, street, pin, zip. Create Address and Employee classes, override equals() in both (with SOPs) and then check by creating 2 employee objects with state and verify if they are equal or not.