Uttara InfoSolutions

www.uttarainfo.com

Collections Practicals 3

1) Test basic methods of Scanner class usage

Steps:
Create 2 Scanner objects wrapping System.in like this (import java.util.* as well):

```
Scanner sc1 = new Scanner(System.in);
Scanner sc2 = new Scanner(System.in);

System.out.println("Enter a string input");
String s1 = sc1.next(); // to read a single word input
//print s1
System.out.println("Enter int as input");
int n = sc1.nextInt();
System.out.println("n = "+n);
double d = sc1.nextDouble();
//print d
String line = sc2.nextLine(); // sentence input
//print line
```

Check out other methods usage of Scanner class.

2) Build a Menu based program to allow the user to

a) Add a friend name
b) List the friend names
c) Remove a friend name
d) Search for friend names given a part string
e) Sort the friend names based on i) alphabetical ii) lengthwise

(See TestMenu.java)
Note: You can pass a Collection reference to another Collection constructor and automatically all the elements will be added from first collection to the second, ex: Set ts = new TreeSet(col); where col is a List for example.

3) Take a sentence and a word as input from the user and

a) print how many occurrences you find of an input word in the sentence (use Collections.frequency() on a List where you add the words in the sentence)
b) sort the sentence i) with duplicates ii) without duplicates and print
c) sort the sentence using string length comparison
d) remove all the occurrences of the word from the sentence

(Go through TestIntvPgms.java for this)

4) Create a Map<String,String>. Test basic working of methods of maps - put(key,val), get(key), remove(key), entrySet(), keySet(), values(), containsKey(), containsValue(). (Look at TestMaps.java)

5) Given a sentence as input, find the number of occurrences of every word in it and print it out (Use Maps).

7) Create a program to take in the country and capitals from the user as inputs. Then a menu should be given to the user to be able to a) search for the capital given a country name b) list the countries and their respective capitals. c) Sort the list output based on country d) *very important* sort the list output based on capitals

8) A Student has a name, date of birth, email and id. Create a menu for the user to allow adding students, get Student info based on id, sort students based on date of birth. Perform date validation correctly for checking date of birth input is a valid date and whether the date is in the past.

9) How sort a Map based on its values?? -> common interview programming question.

10) Create a Stack as an interface with following code

```
public interface Stack
{
	public void push(int element);
	public int pop();
	public int peek();
	public void printElements()
}
```

Create a class called ArrayStack that implements Stack interface like this:

ArrayStack has ints -> 1..n multiplicity which means you can store them in an array instance variable.

```
public class ArrayStack implements Stack
{
	int[] arr = new int[10]; // instance variable where elements will be stored

	int count; // to keep a track of how many elements are filled into the stack
```

```
	public void push(int element)
	{
		1. check if count is < length of array
		2. store element into arr[count].
		3. increment count.
	}
	public int pop()
	{
		1. check if count >0
		2. return arr[count].
		3. decrement count

	}
	public int peek()
	{
		1. check if count > 0
		2. return arr[count]  without decrementing count.
	}
	public void printElements()
	{
		1. Loop over arr count num of times
		2. print each element via SOP
	}
}
```

Build a tester class to test ArrayStack something like this:

```
main(..)
{
	ArrayStack myStack = new ArrayStack();
	myStack.push(5);
	myStack.push(10);
	myStack.push(15);
	myStack.printElements(); -> should print 15 10 5
	SOP(myStack.peek()); -> should print 15
	SOP(myStack.pop());  -> should print 15
	SOP(myStack.peek()); -> should print 10
	myStack.printElements(); -> should print 10 5
	myStack.push(20);
	myStack.printElements(); -> should print 20 10 5

	// test all boundary conditions!
}
```

11) Implement a Stack using LinkedList as its data structure. Use the Stack to implement reversing of words in a given sentence.