CS206 Lab#4: Command Line Arguments, Interfaces, and Exceptions

Exercise 1. Command Line Arguments and importing classes
This exercise very briefly introduces the topic of providing information to your program from the command line.
1. Create a new Project in Eclipse (call it Lab04a)
2.Import the class Main.java from the file `/home/gtowell/Public206/data/lab04` into the project by doing the following:
    a.In the Package Explorer view, click on the > next to Lab04a to expand this item
    b. Right click on "src" then select "import" in the popup menu
    c. Highlight "General > File System"
    d. Tap on Next
    e. In the box to the right of "from directory" enter
        `/home/gtowell/Public206/data/l04`
        then hit "Browse" then hit "open"
    f. A (short) list should appear in boxes below the directory.Put a check next to "Main.java"
    g. Tap on "Finish"
3. Execute the newly imported class from within Eclipse. What is the output?
4. Open a terminal window and cd to the directory containing your project. Assuming you followed class conventions and the directions above execute
```
cd /home/YOU/cs206/Lab04a
```
5. Run your program from the command line as follows:
```
java -cp bin Main
```
    "-cp bin" tells java to find your executable in the bin directory.
6. Run your program again but this time
```
java -cp bin Main a s d f g
```
What is the difference?
7. Run you program one more time
```
java -cp bin Main /home/gtowell/Public206/data/a4/*
```
8. Compare your output to the following unix command which lists every file in a directory
```
ls /home/gtowell/Public206/data/a4
```
    Are 8 and 9 the same? If not why?

Exercise 2: Circular Linked Lists
In this exercise you will import several classes that implement a CircularLinkedList. Then you will fully implement two methods that exists only as stubs in the provided class.

Here are the stubs along with documentation of what they should do.
```
/****
   * @return the number of items in the list
   ****/
  @Override
  public int size()
  {
    return -1;
  }

  /****
   * This methods compare two circular linked lists for "equality".  They are
   * equal under two conditions.  First, they have the same number of items.
   * Second, they have the "same items" in the same order.  "same items"
   * means that the items
```

```
     * are equal according to compareTo, not that the items are the same object.
     * "same order" means that the items have the same order, irrespective of starting
     * point. A list containing "A B C" is in the same order as a list "B C A"
     *  since by changing
     * the starting point of the second it could print as "A B C".  However, "B A C"
     * is not in the same order as "A B C".
     * @return true the two lists have the same number of items in the
     * same order.
     */
    @Override
    public boolean equals(Object object)
    {
       if (!(object instanceof CircularLinkedList))
           return false;  // return false if the item being compared is not a
CircularLinkedList
          CircularLinkedList cll = (CircularLinkedList)object;
       return true;
    }
```

Classes for this exercise are in `/home/gtowell/Public206/data/Lab04/`. You should create a new project then import: Rabbit.java, CircularLinkedList.java, LinkedListInterface.java and CLLMain.java using the procedure described in exercise 1 step 2. Note that in CircularLinkedList.java, the stub functions appear exactly as above.

Finally, extend the main method of CLLMain.java to test your size and equals methods.