Computer Programming
Final exam(2017 Fall)

Checkpoints

1. You should do the Final in your own. You are not allowed to share code with others and/or copy code from other resources. If you are caught, as in the syllabus, you will get a failing grade.
2. Final is open notes, books, and electronic copy notes (including code).
3. No internet is allowed.
    a. SSH and etl are allowed.
4. Grading will be done in Linux environment using java 1.8 and GNU 7.2.
    a. C++ will be compiled with -std=c++11.
5. Program failed to compile/run will result 0.
6. Do not infinite loop your program to repeat unless you are told so.
7. Do not change input/output format unless you are told so.
8. Do not color console.
9. Write your name and student number at top of program as a comment.
10. **Checkin on attendance after you have submitted your file**

Submission

1. Submit your assignment on eTL.
2. Zip your file (or tar) as '<Student ID>-Final.zip'
    a. Do not create subfolder
    b. Do not include package

[1] Circular arrays (Comparison and rotation, Java)

1. In this problem you need to write code to compare and rotate (circular) arrays.
Two arrays A and B are considered to be **equal** if both arrays have the same length, and one of the arrays can be rotated to match all contents of the other.

e.g.) The arrays [-1, 0, 1, 2, -2], [-2, -1, 0, 1, 2], [2, -2, -1, 0, 1], [1, 2, -2, -1, 0] and [0, 1, 2, -2, -1] are all considered equal, as they are all rotations of each other.

(element range: -10000000 ~ +10000000, array size range: 1 ~ 100000,
# of arrays: 1 ~ 100000)

Input format ("_" indicates a blank, last line ends with a newline)

```
[# of arrays]
[size of array1]_[size of array2]_ … _[size of arrayN]
[element 1 of array1]_[element 2]_ … [element n1]
[element 1 of array2]_[element 2]_ … _[element n2]
…
[element 1 of arrayN]_[element 2]_ … _[element nN]
```

Output format (Last line ends with a newline)

```
[# of array pairs(2 arrays) of equal arrays]
```

Input example

```
5
3 3 3 3 3
2 2 3
2 3 2
3 2 3
2 2 3
3 2 2
```

Output

```
6
```

Grading ([input/output file name] means arbitrary input/output file name)

```
javac -encoding utf8 final_01.java
java final_01 [input file name] [output file name]
```

Submission

Compress all your source files for this problem into **final01.zip**

[2] Sorting pairs(C++)

A list of Student records will be given as input. Each student record has two fields: “Name”, and “Grade (max: 100)”. A sorting criteria will also be given as an input. Your program should sort the list of student records according to the criteria specified.

e.g.)
[input]
Sorting criteria: Grade (Increasing)
2
“JeongWonil”, “100”
“YeonilYoo”, “99”
[output]
“YeonilYoo”, “99”
“JeongWonil”, “100”

Sorting of String should be lexicographic order (**ignore cases**) and assume “Name” always contains only English alphabets. The format of input and output will be the following.
(“_” indicates a blank , [format] indicates what kind of output should be there described by “format”)

Input format

```
Sorting criteria:_[standardField]_[Increasing or Decreasing])
[Number of students]
[Name1],_[Grade1]
[Name2],_[Grade2]
…
[NameN],_[GradeN]
```

Output format

```
[sortedName1],_[sortedGrade1]
…
[sortedNameN],_[sortedGradeN]
```

Input example

Sorting criteria: Grade (Decreasing)
4
Alpha, 57
Beta, 33
Gamma, 10
Zeta, 80

Output example

```
Zeta, 80
Alpha, 57
Beta, 33
Gamma, 10
```

Input will be given as a parameter to your program, and also your output file name is in a parameter. Grading will be done like this in command line.

Grade

```
  g++ final_02 -o -std=c++11 final_02.cpp
```

Input will be given at runtime through console.

Submission

Just submit final_02.cpp (and “final_02.h” if you made it)

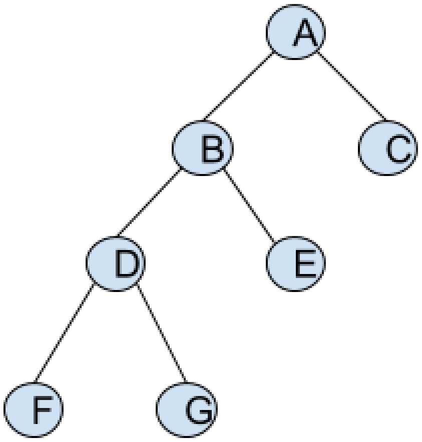
[3]

In this program, you need to implement encoding and decoding methods for binary trees. Let’s first assume that a node in a binary tree has either two children or no children (i.e., there are no nodes with only one child). Each node in the binary tree has a data field whose type is `char’ -- referred to as the label of the node. Assume that the labels of the nodes are unique. The encoding scheme (used in the constructor) can be described using the following procedure:

- Starting at the root node:
  - output the label of the node,
  - encode the left subtree
  - encode the right subtree
  - output label of the node again.

The decoding scheme takes a string (which is an encoded string of a binary tree), reconstructs the corresponding binary tree and returns a pointer to its root.

ex.)



result of encoding = “ABDFFGGDEEBCCA”

Please refer Final3.java and Node.java files provided. **Only submit Final3.java** (Node.java will be moved by TA).

Do not import any java libraries
Do not write extra java classes.
Do not write extra methods other than provided. (you may modify answer method’s return type and parameter.**)**
Failing to following above will result in zero score.