Homework 1

WORKING WITH NUMBER REPRESENTATIONS AND OPERATIONS

VERSION 1.0

The objective of this assignment is to get you comfortable with numbering systems, bitwise operations, and common binary operations. The programming assignment should be implemented in Java. You are required to work alone on this assignment. The assignment accounts for 7.5% towards your cumulative course grade.

This assignment may be modified to clarify any questions (and the version number incremented), but the crux of the assignment and the distribution of points will not change. Any there any changes to the assignment, all changes will be document in the "Change History" section of this assignment,

DUE DATE: Wednesday, February 14th @ 8:00 pm

1 Description of Task

As part of this assignment, you will be responsible for conversions between number representations and bitwise manipulations. The programming assignment should be developed in Java, and your classes must reside in the package cs250.hw1

Your program will be provided with 3 arguments at the command line. You are required to perform a set of operations over these arguments. These operations have been broken up into tasks: the points distribution for each task and the restrictions (and accompanying deductions) are specified in the grading section of this assignment.

Command line execution: java cs250.hw1.Operations <num1> <num2> <num3>

Task 1:
Parse 3 arguments specified as arguments from the command line of the program: one of the arguments will be in decimal, one in binary, and one in hexadecimal. The numbers we specify will be unsigned integers (i.e., the numbers are positive, whole numbers). If an incorrect number of arguments is provided, you should print an error and exit.

Task 2:
Identify the numbering system that each of the arguments refers to. This should occur regardless of the order in which the numbers are specified.

1. A binary number will have a "0b" prefix. For example, the binary number **0b**101011001 represents the decimal number 345
2. A hexadecimal number will have a "0x" suffix. For example, the hexadecimal number 0x159 represents the decimal number 345
3. If either prefix "0b" or "0x" has not been specified, then that number should be considered a decimal number.

Task 3:
Error checking:
1. A binary number should not contain anything other than a 0 or 1.
2. A decimal number should only comprise digits 0 through 9.
3. A hexadecimal number contains digits 0 through 9, and letter A through F either in lowercase or uppercase.

If any of the above validations fail, print an error and exit.

Note: Task 4-8 can be completed using built in java methods. Only use them to check your answer. If you violate these restrictions, points will be deducted (up to 100%) as outlined in the deductions section. You may use these built in java methods only to validate the correctness of your own implementation.

Task 4:
Convert each of the input numbers into the corresponding number in other numbering systems. For example, if the number is in binary, you should convert that number into their corresponding representations in the decimal and hexadecimal numbering systems.

Recall: The nth digit from the right in base b is b^{n-1}
Built in java function that you should not use: `Integer.parseInt(num, radix);`

Task 5:
Compute the 1s complement (in binary) of every number that was specified at the command line. This will require you to first convert every number into binary.

Recall: 1's complement is accomplished by flipping each bit in the binary representation i.e. a 1 becomes a 0, and a 0 becomes a 1.
Built in java function that you should not use: `~num`

Task 6:
Compute the 2s complement (in binary) of every number that was specified at the command line. This will require you to first convert every number into binary.
Recall: 2's comp is accomplished by applying 1's comp then adding 1.
Built in java function that you should not use: `~num + 1`

Task 7:
Compute the bitwise OR, AND, and XOR of the 3 numbers. This will require you to first convert every number into binary.
Recall:
- **OR:** If either of the values is a 1 the result is a 1 otherwise it is a 0.
- **AND:** If both values are 1 the result is 1 otherwise it is a 0.
- **XOR:** If either of the values are 1 and they are not both 1 the value is a 1 otherwise it is a 0.

Built in java function that you should not use: `num1 | num2, num1 & num2, num1 ^ num2`

Task 8:
Compute the bitwise left and right shift of the 3 numbers for 2 shifts. This will require you to first convert every number into binary.
Recall:

- **Left Shift:** This is accomplished by appending an x number of 0's. See lecture slides
- **Right Shift:** Accomplished by moving each digit to the right x times. See lecture slides

Built in java function that you should not use: `num1<<2, num1>>2`

2 Example Outputs

Notes:

- Capitalization and spaces do not matter.
- All numbers are printed in the order provided in the arguments.

Wrong number of arguments:
Command: java cs250.hw1.Operations 1
Output:

```
Task 1
Incorrect number of arguments have been provided. Program Terminating!
```

Invalid characters in the specified numbers:
Command: java cs250.hw1.Operations 15 0b1011 0xhello
Output:

```
Task 1
Correct number of arguments given.
Task 2
15=Decimal
0b1011=Binary
0xhello=Hexadecimal
Task 3
15=true
0b1011=true
0xhello=false
```

Example outputs where the program has been provided with the correct arguments leading to a successful run of the program:
Command: java cs250.hw1.Operations 15 0b1011 0xfa

Output:

```
Task 1
Correct number of arguments given.

Task 2
15=Decimal
0b1011=Binary
0xfa=Hexadecimal
```

```
Task 3
15=true
0b1011=true
0xfa=true

Task 4
Start=15,Binary=0b1111,Decimal=15,Hexadecimal=0xf
Start=0b1011,Binary=0b1011,Decimal=11,Hexadecimal=0xb
Start=0xfa,Binary=0b11111010,Decimal=250,Hexadecimal=0xfa

Task 5
15=1111=>0000
0b1011=1011=>0100
0xfa=11111010=>00000101

Task 6
15=1111=>0001
0b1011=1011=>0101
0xfa=11111010=>00000110

Task 7
1111|1011|11111010=11111111
1111&1011&11111010=00001010
1111^1011^11111010=11111110

Task 8
1111<<2=111100,1111>>2=11
1011<<2=101100,1011>>2=10
11111010<<2=1111101000,11111010>>2=111110
```

3 What to Submit

Use the CS250 *Canvas* to submit a single .zip file that contains:

- Your java source codes, matching this directory structure:
    - cs250
        - hw1
            - Operations.java
- a README.txt file containing a description of each file and any information you feel the TAs need to grade your program.

Filename Convention: The archive file should be named as <FirstName>-<LastName>-HW1.zip . E.g. if you are Cameron Doe and submitting for assignment 1, then the zip file should be named Cameron-Doe-HW1.zip.

4 Grading

The assignments must compile and function correctly on machines in the CSB-120 Lab. Assignments that work on your laptop on your particular flavor of Linux, but not on the Lab machines are considered unacceptable.

This assignment will contribute a maximum of 7.5 points towards your final grade. The grading will also be done on a 7.5 point scale. The points breakdown is as follows:

1 point each for correctly performing Tasks 1, 2, 3, 4, 5, 6 and 7 (**7 points**)

0.5 point for Task 8

Deductions:
There is a 7.5-point deduction (i.e., you will have a zero on the assignment) if you:
(1) Build a GUI

Use of the following methods will result in a 100% deduction of the task points where used:
(1) Integer.parse
(2) Integer.toHexString
(3) Integer.toBinaryString
(4) Built in bitwise operations (~, ^, |, &, <<, >>)
    a. Note: Logical operations are allowed (!, ||, &&, etc.)

A 100% deduction will occur from using any of the methods in this list (and all their overloaded variants). *Please note that 1) Integer.parse refers to Integer.parseInt*. An additional set of overloaded methods has been included to the deductions list from the Integer class: ***Integer.valueOf***

These methods are illegal as they remove the need for students to perform parsing followed by conversions.

You are required to **work alone** on this assignment.

5 Late Policy

Please check the class policy on submitting late assignments. You are allowed to submit assignments up to 2 days with a per-day deduction of 7.5%.

6 Version Change History

This section will reflect the change history for the assignment. It will list the version number, the date it was released, and the changes that were made to the preceding version. Changes to the first public release are made to clarify the assignment; the spirit or the crux of the assignment will not change.

Version	**Date**	**Comments**
1.0	1/23/2024	First public release of the assignment.