CMPE 230 Systems Programming

Project (due Oct. 30th)

(This project can be implemented in groups of two students.)

(Use C/C++ or Java language to implement the project)

In this project, you will implement a simple compiler called BITC that generates A86 code for bitwise operators. BITC will accept (i) expressions and (ii) assignment statements. Expressions will be infix expressions involving bitwise & and | operations. There will also be other bitwise operations: xor, complement, shift and rotate. All operations are shown in the table below:

Operation	**Meaning**
a & b	Returns bitwise a and b.
a | b	Returns bitwise a or b.
xor(a,b)	Returns bitwise a xor b.
ls(a,i)	Returns i left shift(s) of a.
rs(a,i)	Returns i right shift(s) of a.
lr(a,i)	Returns i left rotation(s) of a.
rr(a,i)	Returns i right rotation(s) of a.
not(a)	Returns bitwise complement of a.

You can assume all values and results of operations will be 16 bit values. An example of BITC usage are given below:

BITC compiler

Suppose the file **example.bc** contains:

```
$x = abcd
$y = 1
$x = not($x | xor($x,f123))
$y = not(ls($y,2))
$x = xor($x,$y)
$y = xor($x,$y)
$x = xor($x,$y)
$x
$y
```

```
%bitc example.bc
 example.asm was generated.
%a86 example.asm
%example
fffb
0410
```

Please note the following:

- You can assume that an undefined variable has value 0.
- Variable names start with $.
- All variables and expressions are 16 bit.
- All numbers are expressed in hexadecimal format.
- The operations can accept expressions.
- & has higher precedence than | .

Grading

Your project will be graded according to the following criteria:

Documentation (written document describing how you implemented your project)	12%
Comments in your code	8%
Implementation and tests	80%

Late Submission

If the project is submitted late, the following penalties will be applied:

- 0 < hours late <= 24 : 25%
- 24 < hours late <= 48 : 50%
- 48 < hours late <= 72 : 75%
- hours late > 72 : 100%

Proof of Existence

Before you submit your project, please timestamp (notarize) your project zip file at https://opentimestamps.org/ Do *NOT* lose the .ots timestamp file and the submitted project zip file. The .ots timestamp file is a proof that your project zip file existed during the time of submission. If for some reason something went wrong with submission, the .ots file and the corresponding project zip file will prove that your project zip file existed at the time of submission. Only the project zip file that matches the .ots will be accepted. If you show a project zip file that does not match (correspond to) the .ots file, it will *NOT* be accepted !.