Homework Assignment #1 (Posted on 3/15, Due 4/12)

In Chapter 2, we have learned how to build a simple compiler for the *AC* language. Now you may download this compiler (coded in C) from the class website on CEIBA.

Files for this assignment are available in the following directories:
/src contains the C source files and a makefile for building the *AC* compiler,
/test contains a set of sample tests.
You may create the simple compiler as follows:

```
    cd src
    make
```

and the AcDc compiler will be generated, you may test it using sample test files in the test directory.

```
    ./AcDc ../test/sample1.ac output1
    ./AcDc ../test/sample2.ac output2
```

In this assignment, you are required to extend the AcDc compiler in three ways:

1. **Extend the AC language to accept integer multiply (*) and divide (/) operators. You must correctly handle the precedence of * and / operators, which are higher than the + and - operators.**
2. **The AC language supports only single character variable names. A real programming language would allow for longer names. You are required to relax this restriction. Note that in the test data, the length of a variable name will not exceed 64 characters, and the number of different variables will not exceed 23 (to simplify later code generation).**
3. **Enhance the AcDc compiler with a simple optimization called "constant folding", which evaluates constant expressions at compile time.**

For example, the following expression
a = 10 + 20 – 5 + b

could be turned into
a = 25 + b

With constant folding at compile time, fewer instructions would be generated. Note that you are **NOT** required to exploit the constant folding opportunities in the following expression:
a – 100 – 50 + 6

This is because the order of evaluation for the above expression is actually
(((a - 100) – 50) + 6)
Therefore, there are no constant expressions available for folding unless more complicated optimizations such as applying the commutative laws to this expression.

When integer and float constants are mixed in expressions, you need to pay attention to the correctness of constant folding, for example, 1 / 2 = 0, but 1.0 / 2 = 0.5.

Submission requirements:

1) DO NOT change the executable name (AcDc).
2) Use the script file "tar.sh" to package your assignment into a single file. Then upload your packaged assignment to CEIBA.

```
Usage: ./tar.sh source_directory studentID version_number
Example: ./tar.sh hw1 r08921059 ver1
Output: r08921059_ver1.tar.bz2 (submit this file)
```

3) We grade the assignments on linux.

TAs' email addresses are as follows:
r08921059@ntu.edu.tw

If you need to make changes to your submitted files, you may submit a new version before the deadline.