Homework Assignment #1 (Posted on 10/03, Due 10/16)

In Chapter 2, we have learned how to build a simple compiler for the *AC* language. Now you may download this compiler (written 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 compile 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. You are required to lift this restriction and allow for variable length names. Note that in the test data, the length of a variable name will not exceed 256 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

should be turned into
a = 25 + b

```
With constant folding at compile time, fewer instructions would be
generated for DC.


Note that you are not required to exploit the constant folding
opportunities in the following expressions:
a – 100 –50 + 6


Because the order of evaluation for the above expression is actually
(((a-100) – 50) + 6)
Therefore, there are no constant expressions to be folded 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 studentID1_studentID2 (all student IDs in your team) version_number

Example: ./tar.sh hw1 12345_12346 ver1

Output: 12345_12346_ver1.tar.bz2 (submit this file)

3) We grade the assignments on the linux1 server (linux1.csie.org). Before summiting your assignment, you should make sure your version works correctly on linux1.

Use a separate e-mail to inform TAs about the students in your group.

TAs' email addresses are as follows:

施詠翔

r02922036@csie.ntu.edu.tw

劉建旻

jm410287@gmail.com

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