# What a Compiler does - $\operatorname{Exercises}^*$

## Charles Severance Kevin Dowd

This work is produced by OpenStax-CNX and licensed under the Creative Commons Attribution License  $3.0^{\dagger}$ 

### Exercise 1

Does your compiler recognize dead code in the program below? How can you be sure? Does the compiler give you a warning?

```
main()
{
    int k=1;
    if (k == 0)
        printf ("This statement is never executed.\n");
}
```

#### Exercise 2

Compile the following code and execute it under various optimization levels.

Try to guess the different types of optimizations that are being performed to improve the performance as the optimization is increased.

```
REAL*8 A(1000000)
D0 I=1,1000000
    A(I) = 3.1415927
ENDD0
D0 I=1,1000000
    A(I) = A(I) * SIN(A(I)) + COS(A(I)) ENDD0
PRINT *,"All Done"
```

#### Exercise 3

Take the following code segment and compile it at various optimization levels. Look at the generated assembly language code (-S option on some compilers) and find the effects of each optimization

<sup>\*</sup>Version 1.3: Aug 25, 2010 11:20 am -0500

 $<sup>^{\</sup>dagger} http://creativecommons.org/licenses/by/3.0/$ 

level on the machine language. Time the program to see the performance at the different optimization levels. If you have access to multiple architectures, look at the code generated using the same optimization levels on different architectures.

```
REAL*8 A(1000000)
COMMON/BLK/A
.... Call Time
DO I=1,1000000
A(I) = A(I) + 1.234
ENDDO
.... Call Time
END
```

Why is it necessary to put the array into a common block?