CSC301 Assignment #3

Due Wednesday 9/28 in Sakai.

Problem 4.4

Exercise 4  Suppose the target assembly language for a compiler has these five instructions for integers:

\[
\begin{align*}
\text{load} & \quad \text{address}, \text{reg} \\
\text{add} & \quad \text{reg}, \text{reg}, \text{reg} \\
\text{sub} & \quad \text{reg}, \text{reg}, \text{reg} \\
\text{mul} & \quad \text{reg}, \text{reg}, \text{reg} \\
\text{store} & \quad \text{reg}, \text{address}
\end{align*}
\]

In these instructions, an \textit{address} is the name of a static variable (whose actual address will be filled in by the loader). A \textit{reg} is the name of an integer register, a special extra-fast memory location inside the processor. The target assembly language has three integer registers: \texttt{r1}, \texttt{r2}, and \texttt{r3}. The \texttt{load} instruction loads the integer from the given memory address into the given register. The \texttt{add} instruction adds the second register to the first register and places the result in the third register. The \texttt{sub} instruction subtracts the second register from the first register and places the result in the third register. The \texttt{mul} instruction multiplies the first register by the second register and places the result in the third register. The \texttt{store} instruction stores the integer from the given register at the given memory address. So, for example, the compiler might translate the assignment \texttt{result := offset+(width*n) into this:}

\[
\begin{align*}
\text{load width,} & \quad \text{r1} \\
\text{load } & \quad \text{n,} \quad \text{r2} \\
\text{mul } & \quad \text{r1,} \quad \text{r2,} \quad \text{r1} \\
\text{load offset,} & \quad \text{r2} \\
\text{add } & \quad \text{r2,} \quad \text{r1,} \quad \text{r1} \\
\text{store } & \quad \text{r1,} \quad \text{result}
\end{align*}
\]

Using this assembly language, give translations of the following assignment statements. Use as few instructions as possible.

\begin{itemize}
  \item \textbf{a.} net := gross - costs
  \item \textbf{b.} volume := (length * width) * height
  \item \textbf{c.} cube := (x * x) * x
  \item \textbf{d.} final := ((a - abase) * (b - bbase)) * (c - cbase)
\end{itemize}

(This is an example of a \textit{load/store architecture}. Many modern microprocessors implement an architecture like this, though usually with more registers.)