Question 2 (15 points):

In an important application 20% of the instructions are branches. When this application is running on a pipelined processor, 30% of the total cycles executed are wasted as stall cycles because of branch miss predictions. Currently this applications takes 10 minutes to execute on a processor running with a clock frequency of 4 GHz.

a. (**5 points**) How many cycles are wasted because of the stalls caused by branch miss predictions when this application executes?

$$\begin{aligned}
\text{Clock Cycle} &= \frac{1}{4 \times 10^9 \text{ Hz}} = 0.25 \times 10^{-9} \text{ seconds} \\
\text{Stall time} &= 30\% \times 600 \text{ seconds} = 180 \text{ seconds} \\
\#\text{ stall cycles} &= \frac{180 \text{ seconds}}{0.25 \times 10^9 \frac{\text{seconds}}{\text{cycle}}} = 720 \times 10^9 \text{ cycles}
\end{aligned} \tag{1}$$

b. (**5 points**) The compiler team got a better understanding of the pipelined architecture and of the branch prediction mechanism implemented by the hardware. They were able to eliminate 2/3 of the stalls due to the branch miss prediction. What percentage of the execution time is now wasted in stall cycles because of miss predictions?

In the original code, 3 minutes out of 10 minutes were spent in stalls and 7 minutes were spent in the rest of the execution. Now 1 minute is spent in stalls and the application takes 8 minutes to execute. Therefore:

$$\%\text{ of stall time} = \frac{1 \text{ second}}{8 \text{ seconds}} \times 100 = 12.5\%$$

c. (**5 points**) What is the speedup that results from this improvement to the code?

$$\text{Speedup} = \frac{\text{original time}}{\text{improved time}} = \frac{10}{8} = 1.25 \text{ times}$$