The DiBlas7777 has come up with a small benchmark to predict their pipelining and caching system. Consider the following code.

```
add $10, $0, $0
addi $15, $0, 100

begin:
    lw $11, 0($7)
    add $11, $10, $11
    sw $11, 0($7)
    lw $13, 0($7)
    lw $12, 4($7)
    add $13, $11, $12
    sw $13, 4($7)
    addi $10, $10, 1
    beq $10, $15, endloop
    addi $7, $7, 8
    j begin

endloop:
    add $15, $0, $0
```
(4 points) How long will this code take to execute the code on the DiBlas7777, assuming the pipelined CPU studied in class running at 800MHz with a hazard detection unit, a forwarding unit, one branch delay slot, and a static branch predictor set to “predict not taken?” Do not rearrange, modify, or rewrite the code!

(4 points) Because of the fine job you have done describing the pipelined DiBlas7777, you have been transferred to the memory division.

The DiBlas7777 ships with 32MB of main memory. Consider a direct-mapped, write-back, 32-block data cache with 4-byte blocks, and a separate yet identical instruction cache. Initially, the cache is cold (i.e., it starts out empty and unaccessed). The cache miss penalty is 10 clock cycles. How long will the code take to execute?
(1 point) Is there measured improvement if the DiBlas7777 were to use a 2-way set-associative cache? How much?

(1 point) Is there measured improvement if the DiBlas7777 were to use a fully-associative cache? How much?