5.2.2 Program Instructions [20 points]

Fill in the blanks below with the six-instruction sequence in program order. When referring to registers, please use their architectural names (R0 through R9). Place the register with the smaller architectural name on the left source register box.
For example, `ADD R8 ⇐ R1, R5.` (20 points if everything is correct.)

MUL	R2	⇐	R3	,	R9
MUL	R8	⇐	R4	,	R7
ADD	R9	⇐	R2	,	R8
MUL	R7	⇐	R7	,	R9
MUL	R1	⇐	R1	,	R7
ADD	R5	⇐	R1	,	R9

6 GPUs and SIMD

We define the *SIMD utilization* of a program run on a GPU as the fraction of SIMD lanes that are kept busy with *active threads* during the run of a program. As we saw in lecture and practice exercises, the SIMD utilization of a program is computed across the *complete run* of the program.

The following code segment is run on a GPU. Each thread executes **a single iteration** of the shown loop. Assume that the data values of the arrays A, B, and C are already in vector registers so there are no loads and stores in this program. (Hint: Notice that there are 6 instructions in each thread.) A warp in the GPU consists of 64 threads, and there are 64 SIMD lanes in the GPU. Please assume that all values in array B have magnitudes less than 10 (i.e., $|B[i]| < 10$, for all i).

```
for (i = 0; i < 1024; i++) {
    A[i] = B[i] * B[i];
    if (A[i] > 0) {
        C[i] = A[i] * B[i];
        if (C[i] < 0) {
            A[i] = A[i] + 1;
        }
        A[i] = A[i] - 2;
    }
}
```

Please answer the following five questions.

(a) [5 points] How many warps does it take to execute this program?

Warps = (Number of threads) / (Number of threads per warp)
Number of threads = 2^{10} (i.e., one thread per loop iteration).
Number of threads per warp = 64 = 2^6 (given).
Warps = $2^{10}/2^6 = 2^4$

(b) [5 points] What is the maximum possible SIMD utilization of this program?

100%