Hadassah Academic College
Department of Computer Science

Advanced Computer Architecture Exercise 5

1. Consider the following code fragment:

```
 1        ADDI R2, R0, #0x1000
 2 L1:    LW R20, 4000(R2)
 3 L2:    SUBI R1, R2, #0x0004
 4        LW R10, 4000(R1)
 5        SGT R4, R10, R20
 6        BEQZ R4, L3
 7        SW 4000(R1), R20
 8        SW 4000(R2), R10
 9 L3:    BNEZ R1, L2
10        SUBI R2, R2, #0x0004
11        BNEZ R2, L1
12 STOP:  ADD R31, R20, R29
```

According to the trace cache method, each entry begins with the target or fall-through for a branch instruction (including branches between instructions within a trace) and ends with a branch instruction. For the code above (from exercise 2) the first trace T1 is given in the table:

T1	9	3	4	5	6		
T2							
T3							

A. Briefly explain how this trace is built during execution.

B. At the second execution of line 6, R20 > R10. Briefly explain how the code continues.

C. At the third execution of line 6, R20 < R10. The trace cache builds a new trace from the fall-through up to (and including) line 6. Write the new trace T2.

D. For the next ten executions of line 6, R20 < R10 occurs 5 times in a row, and then R20 > R10 occurs 5 times. Write the execution as a sequence of traces.

E. In the last iteration of the inner loop, both T1 and T2 are mispredictions. The trace cache builds a new trace from the fall-through of line 9 up to (and including) line 6. Write the new trace T3.

F. Toward the end of the second out loop iteration, there is a strong prediction of R20 > R10. Write the sequence of traces from the last trace of the inner loop to the first trace of the next execution of the inner loop.

2. A. Rewrite the code fragment using predication where appropriate.

B. Explain briefly how predication simplifies the trace cache