Question 4 (20 points):

```
1  vecAdd1: add   a0, zero, zero
2           li    t1, 0x00008000
3           li    t2, 0x00009000
4  next:    lw    t3, 0(t1)
5           add   a0, a0, t3
6           addi  t1, t1, 4
7           bne   t1, t2, next
8           jalr  zero, ra, 0
```

(a) Code for `vecAdd1`

```
10 vecAdd2: add   a0, zero, zero
11          li    t1, 0x00008000
12          li    t2, 0x0000C000
13 next:    lw    t3, 0(t1)
14          add   a0, a0, t3
15          addi  t1, t1, 16
16          bne   t1, t2, next
17          jalr  zero, ra, 0
```

(b) Code for `vecAdd2`

Figure 1: Two versions of a function that sum elements of a vector.

Figure **??** shows two versions of a RISC-V code that returns the sum of some elements of a vector. Both versions of this code are executed in a processor with a 16KB L1 Data Cache with 16-byte cache blocks.

1. (**5 points**) Assume that this is a 32-bit address machine. How many elements of the vector are accessed by `vecAdd1` and how many elements of the vector are accessed by `vecAdd2`?

2. (**5 points**) If the L1 Data Cache is directly mapped, what is the hit ratio for the L1 Data Cache for `vecAdd1` and for `vecAdd2`?

3. (**5 points**) What is the effect in the hit ratios if the L1 Data Cache retains the same total data storage of 16KB but is made two-way set associative?

4. (**5 points**) This machine has a 32-bit address bus and a two-way set associative L1 Data Cache. How many bits are used for each of the following components of a data-cache access?

   - Offset:
   - Index:
   - Tag: